Net-Centric Implementation

Part 2: Traceability

v3.1.0

22 December 2009

Net-Centric Enterprise Solutions for Interoperability (NESI) is a collaborative activity of the USN PEO for C4I and Space, the USAF Electronic Systems Center, and the Defense Information Systems Agency.

Approved for public release; distribution is unlimited.

SSIC: 3093.4

NESI-X Subdocument generated using: **View, Part 2: Traceability**
Generated: **Wed Dec 23 11:09:30 PST 2009**
NESI-X Version: **v2.0.10 build 5995 - 2009/12/15 13:40**

Table of Contents

Perspectives 16

NESI Executive Summary 16

Part 2: Traceability 18

ASD(NII): Net-Centric Guidance 19

Data 20

Design Tenet: Make Data Visible 21

Design Tenet: Make Data Accessible 22

Design Tenet: Make Data Understandable 24

Design Tenet: Make Data Trustable 26

Design Tenet: Make Data Interoperable 27

Design Tenet: Provide Data Management 29

Design Tenet: Be Responsive to User Needs 30

Services 31

Design Tenet: Service-Oriented Architecture (SOA) 32

Design Tenet: Open Architecture 38

Design Tenet: Scalability 42

Design Tenet: Availability 44

Design Tenet: Accommodate Heterogeneity 45

Design Tenet: Decentralized Operations and Management 48

Design Tenet: Enterprise Service Management 49

Information Assurance/Security 50

Design Tenet: Net-Centric IA Posture and Continuity of Operations 51

Design Tenet: Identity Management, Authentication, and Privileges 52

Design Tenet: Mediate Security Assertions 57

Design Tenet: Cross-Security-Domains Exchange 58

Design Tenet: Encryption and HAIPE 59

Design Tenet: Employment of Wireless Technologies 60

Other Design Tenets 61

Transport 63

Design Tenet: IPv6 64
Design Tenet: Packet Switched Infrastructure 66
Design Tenet: Layering and Modularity 67
Design Tenet: Transport Goal 68
Design Tenet: Network Connectivity 70
Design Tenet: Concurrent Transport of Information Flows 71
Design Tenet: Differentiated Management of Quality-of-Service 72
Design Tenet: Inter-Network Connectivity 73
Design Tenet: Joint Technical Architecture [now DISR] 74
Design Tenet: RF Acquisition 75
Design Tenet: Joint Net-Centric Capabilities 76
Design Tenet: Operations and Management of Transport and Services 78
Open Technology Development 81
Open Architecture 82
Open Standards 84
Open Development Collaboration 85
Open Source (Software) 86
Open Systems 87
Naval Open Architecture 88
Interoperability 89
Maintainability 94
Extensibility 98
Composeability 99
Reusability 100
Relationship with the JCIDS Process 102
DISR Service Areas 104
C4ISR: Payload Platform 105
Standard Interface Documentation 106
Implement a Component-Based Architecture 108
Public Interface Design 109
Communications Applications 110
Software Communication Architecture 111

Network Information Assurance 112
Node Transport 113
Physical and Data Link Layers 114
Network Layer 116
Internet Protocol (IP) 117
IPv4 to IPv6 Transition 119
IP Routing and Routers 121
Integration of Non-IP Transports 124
Transport Layer 125
Subnets and Overlay Networks 126
Broadcast, Multicast, and Anycast 128
Virtual Private Networks (VPN) 129
Ad Hoc Networks 130
Network Services 131
Domain Name System (DNS) 133
Dynamic Host Configuration Protocol (DHCP) 135
Network Time Service 138
Application Layer Protocols 140
Mobility 142
Traffic Management 144
Planning Network Services 145
Architectural Approaches to Traffic Management 146
Traffic Engineering 148
Text Conferencing 150
Data Interchange Services 152
Messaging 153
Message-Oriented Middleware (MOM) 154
Data Distribution Service (DDS) 156
Decoupling Using DDS and Publish-Subscribe 161
DDS Quality of Service 162
DDS Data-Centric Publish-Subscribe (DCPS) 164
DDS Domains - Global Data Spaces 166

Reading/Writing Objects within a DDS Domain 168
Messaging within a DDS Domain 170
DDS Data Local Reconstruction Layer (DLRL) 171
Messaging with MSMQ 172
Web Services 173
SOAP 175
Web Services Compliance 177
WSDL 178
Insulation and Structure 179
Universal Description, Discovery, and Integration (UDDI) 180
Service Definition Framework 182
CORBA 188
Data Distribution Service (DDS) 156
Decoupling Using DDS and Publish-Subscribe 161
DDS Quality of Service 162
DDS Data-Centric Publish-Subscribe (DCPS) 164
DDS Domains - Global Data Spaces 166
Reading/Writing Objects within a DDS Domain 168
Messaging within a DDS Domain 170
DDS Data Local Reconstruction Layer (DLRL) 171
Data 190
XML 192
XML Syntax 193
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202

XML Processing 203
XPath 204
XSLT 205
Parsing XML 207
XML Validation 208
Metadata Registry 209
Data Modeling 211
Metadata 214
Relational Database Management Systems 216
Net-Centric Information Engineering 218
Node Data Strategy 219
Data Management Services 221
DDS Data Local Reconstruction Layer (DLRL) 171
Relational Database Management Systems 216
Data 190
XML 192
XML Syntax 193
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202
XML Processing 203
XPath 204
XSLT 205
Parsing XML 207
XML Validation 208
Metadata Registry 209

Data Modeling 211
Metadata 214
Relational Database Management Systems 216
Distributed Computing Services 222
Standard Interface Documentation 106
Implement a Component-Based Architecture 108
Public Interface Design 109
Messaging 153
Message-Oriented Middleware (MOM) 154
Data Distribution Service (DDS) 156
Decoupling Using DDS and Publish-Subscribe 161
DDS Quality of Service 162
DDS Data-Centric Publish-Subscribe (DCPS) 164
DDS Domains - Global Data Spaces 166
Reading/Writing Objects within a DDS Domain 168
Messaging within a DDS Domain 170
DDS Data Local Reconstruction Layer (DLRL) 171
Messaging with MSMQ 172
Web Services 173
SOAP 175
Web Services Compliance 177
WSDL 178
Insulation and Structure 179
Universal Description, Discovery, and Integration (UDDI) 180
Service Definition Framework 182
.NET Framework 223
CORBA 188
Data Distribution Service (DDS) 156
Decoupling Using DDS and Publish-Subscribe 161
DDS Quality of Service 162
DDS Data-Centric Publish-Subscribe (DCPS) 164
DDS Domains - Global Data Spaces 166

Reading/Writing Objects within a DDS Domain 168
Messaging within a DDS Domain 170
DDS Data Local Reconstruction Layer (DLRL) 171
Net-Centric Information Engineering 218
Enterprise Service Bus (ESB) 224
Environment Management 227
Implement a Component-Based Architecture 108
Public Interface Design 109
Software Communication Architecture 111
Enterprise Management 228
Standard Interface Documentation 106
Services 237
Core Enterprise Services (CES) 240
Overarching CES Issues 242
CES Definitions and Status 243
CES and Intermittent Availability 244
Cross-Domain Interoperation 245
Net-Ready Key Performance Parameter (NR-KPP) 246
Information Assurance (IA) 247
Net-Centric Operations and Warfare Reference Model (NCOW RM) 248
Key Interface Profile (KIP) 249
Integrated Architectures 250
NCES Directory Services 251
Service Discovery 252
NCES Federated Search 253
Collaboration Services 254
Text Conferencing 150
Service Enablers 255
Service Discovery 252
Information Exchange Patterns 256
Service Optimization and Scalability 257
Utility Services 259

Internationalization Services 261
Data Modeling 211
Designing User Interfaces for Internationalization 262
Operating System Services 263
Software Communication Architecture 111
Software Security 264
Technologies and Standards for Implementing Software Security 265
Public Key Infrastructure (PKI) and PK Enable Applications 266
Key Management 267
Certificate Processing 268
Smart Card Logon 270
XML Digital Signatures 271
Encryption Services 273
SOAP Security 274
Security Assertion Markup Language (SAML) 276
RDBMS Security 277
LDAP Security 278
JNDI Security 279
Application Resource Security 280
Java Security 281
Policies and Processes for Implementing Software Security 282
Secure Coding and Implementation Practices 283
Apply Principle of Least Privilege 284
Practice Defense in Depth 285
Apply Secure Coding Standards 286
Apply Quality Assurance to Software Development 287
Validate Input 288
Heed Compiler Warnings 289
Handle Exceptions 290
Data at Rest 291
Mobile Code 292
Security Services 295

Software Security 264
Technologies and Standards for Implementing Software Security 265
Public Key Infrastructure (PKI) and PK Enable Applications 266
Key Management 267
Certificate Processing 268
Smart Card Logon 270
XML Digital Signatures 271
Encryption Services 273
SOAP Security 274
Security Assertion Markup Language (SAML) 276
RDBMS Security 277
LDAP Security 278
JNDI Security 279
Application Resource Security 280
Java Security 281
Policies and Processes for Implementing Software Security 282
Secure Coding and Implementation Practices 283
Apply Principle of Least Privilege 284
Practice Defense in Depth 285
Apply Secure Coding Standards 286
Apply Quality Assurance to Software Development 287
Validate Input 288
Heed Compiler Warnings 289
Handle Exceptions 290
Data at Rest 291
Mobile Code 292
Enterprise Security 296
Cryptography 297
Integrity 299
Computing Infrastructure Integrity 303
Network Infrastructure Integrity 306
User Environment Integrity 314

Data, Application and Service Integrity 315
Identity Management 316
Public Key Infrastructure 317
Authorization and Access Control 319
Confidentiality 321
Black Core 322
Network Information Assurance 112
Trusted Guards 323
Network Information Assurance 112
User Interface Services 324
User Interfaces 325
Human-Computer Interaction 326
Designing User Interfaces for Internationalization 262
Designing User Interfaces for Accessibility 328
Human Factor Considerations for Web-Based User Interfaces 329
Browser-Based Clients 332
XML Rendering 333
Active Server Pages (ASP) 334
Active Server Pages for .NET (ASP.NET) 335
Java Server Pages (JSP) 336
Web Portals 337
Style Sheets 338
Thick Clients 339
Human-Computer Interaction 326
Designing User Interfaces for Internationalization 262
Designing User Interfaces for Accessibility 328
Human Factor Considerations for Web-Based User Interfaces 329
User (Physical/Cognitive) 340
Human-Computer Interaction 326
Designing User Interfaces for Internationalization 262
Designing User Interfaces for Accessibility 328
Human Factor Considerations for Web-Based User Interfaces 329

Exposure Verification Tracking Sheets 341
Data Exposure Verification Tracking Sheet 342
Data Visibility 343
Design Tenet: Make Data Visible 21
Design Tenet: Provide Data Management 29
Net-Centric Data Strategy (NCDS) 344
Metadata Registry 209
Data Accessibility - Policy 346
Net-Centric Data Strategy (NCDS) 344
Data Accessibility - Operational 347
NCES Federated Search 253
Net-Centric Data Strategy (NCDS) 344
Data Understandability 348
Net-Centric Data Strategy (NCDS) 344
Metadata Registry 209
Design Tenet: Make Data Understandable 24
Data Modeling 211
Metadata 214
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202
Service Exposure Verification Tracking Sheet 349
Service Visibility - Registered 350
Service Definition Framework 182
Service Enablers 255
Service Discovery 252

Information Exchange Patterns 256
Metadata Registry 209
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202
WSDL 178
Service Visibility - Discoverable 351
Universal Description, Discovery, and Integration (UDDI) 180
Metadata Registry 209
WSDL 178
Service Definition Framework 182
Service Enablers 255
Service Discovery 252
Information Exchange Patterns 256
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202
Service Accessibility - Policy 352
Metadata Registry 209
Design Tenet: Identity Management, Authentication, and Privileges 52

Service Accessibility - Registered 353
Metadata Registry 209
Universal Description, Discovery, and Integration (UDDI) 180
WSDL 178
Service Definition Framework 182
Service Enablers 255
Service Discovery 252
Information Exchange Patterns 256
Service Understandability - Registered 354
Metadata Registry 209
Metadata 214
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202
Service Understandability - COI Data Models 355
Metadata Registry 209
Metadata 214
XML Semantics 194
XML Schema Documents 195
Using XML Substitution Groups 196
Defining XML Types 197
XML Schema Files 198
Using XML Namespaces 199
Defining XML Schemas 200
Versioning XML Schemas 201
XML Instance Documents 202

Data Modeling 211
Guidance and Best Practice Details 542
Glossary 932
References 994

P1117: NESI Executive Summary

Net-Centric Enterprise Solutions for Interoperability (***NESI***) provides actionable guidance for acquiring net-centric solutions that meet DoD **Network Centric Warfare** goals. The concepts in various directives, policies and mandates, such as those included in the References section of this perspective, are the basis of NESI guidance. The NESI *Net-Centric Implementation* documentation does the following: addresses architecture, design and implementation; provides compliance checklists; and includes a collaboration environment with a repository.

NESI is a body of architectural and engineering knowledge that helps guide the design, implementation, maintenance, evolution, and use of **Information Technology** (**IT**) in net-centric solutions for military application. NESI provides specific technical recommendations that a DoD organization can use as references. NESI serves in many areas as a reference set of compliant instantiations of DoD directives, policies and mandates.

NESI is derived from a studied examination of enterprise-level needs and from the collective practical experience of recent and on-going program-level implementations. NESI is based on current and emergent technologies and describes the practical experience of system developers within the context of a minimal top-down technical framework. NESI guidance strives to be consistent with commercial best practices in the area of enterprise computing and IT.

NESI applies to all phases of the acquisition process as defined in DoD Directive 5000.1 [R1164] and DoD Instruction 5000.2; [R1165] NESI provides explicit guidance for implementing net-centricity in new acquisitions and for migrating legacy systems to greater degrees of net-centricity.

NESI subsumes a number of references and directives; in particular, the Air Force C2 Enterprise Technical Reference Architecture (C2ERA) and the Navy Reusable Applications Integration and Development Standards (RAPIDS). Initial authority for NESI is per the Memorandum of Agreement between Commander, Space and Naval Warfare Systems Command (SPAWAR); Navy Program Executive Officer, C4I & Space (now PEO C4I); and the United States Air Force Electronic Systems Center (ESC), dated 22 December 2003, Subject: Cooperation Agreement for Net-Centric Solutions for Interoperability (NESI). The Defense Information Systems Agency (DISA) formally joined the NESI effort in 2006.

Perspectives	NESI ***Perspectives*** describe a topic and encompass related, more specific Perspectives or encapsulate a set of Guidance and Best Practice details, Examples, References, and Glossary entries that pertain to the topic.
Guidance	NESI ***Guidance*** is in the form of atomic, succinct, absolute and definitive Statements related to one or more Perspectives. Each Guidance Statement is linked to Guidance Details which provide Rationale, relationships with other Guidance or Best Practices, and Evaluation Criteria with one or more Tests, Procedures and Examples which facilitate validation of using the Guidance through observation, measurement or other means. Guidance Statements are intended to be binding in nature, especially if used as part of a Statement of Work (SOW) or performance specification.
Best Practices	NESI ***Best Practices*** are advisory in nature to assist program or project managers and personnel. Best Practice Details can have all the same parts as NESI Guidance. The use of NESI Best Practices are at the discretion of the program or project manager.
Examples	NESI ***Examples*** illustrate key aspects of Perspectives, Guidance, or Best Practices.
Glossary	NESI ***Glossary*** entries provide terms, acronyms, and definitions used in the context of NESI Perspectives, Guidance and Best Practices.
References	NESI ***References*** identify directives, instructions, books, Web sites, and other sources of information useful for planning or execution.

Releasability Statement

NESI *Net-Centric Implementation* v3.1 is cleared for public release by competent authority in accordance with DoD Directive 5230.9; [R1232] *Distribution Statement A: Approved for public release; distribution is unlimited* applies to the documentation set. Obtain electronic copies of this document at http://nesipublic.spawar.navy.mil.

Vendor Neutrality

NESI documentation sometimes refers to specific vendors and their products in the context of examples and lists. However, NESI is vendor-neutral. Mentioning a vendor or product is not intended as an endorsement, nor is a lack of mention intended as a lack of endorsement. Code examples typically use open-source products since NESI is built on the open-source philosophy. NESI accepts inputs from multiple sources so the examples tend to reflect contributor preferences. Any products described in examples are not necessarily the best choice for every circumstance. Users are encouraged to analyze specific project requirements and choose tools accordingly. There is no need to obtain, or ask contractors to obtain, the tools that appear as examples in this guide. Any lists of products or vendors are intended only as examples, not as a list of recommended or mandated options.

Disclaimer

Every effort has been made to make NESI documentation as complete and accurate as possible. Even with frequent updates, this documentation may not always immediately reflect the latest technology or guidance. Also, references and links to external material are as accurate as possible; however, they are subject to change or may have additional access requirements such as Public Key Infrastructure (PKI) certificates, Common Access Card (CAC) for user identification, and user account registration.

Contributions and Comments

NESI is an open project that involves the entire development community. Anyone is welcome to contribute comments, corrections, or relevant knowledge to the guides via the Change Request tab on the NESI Public site, http://nesipublic.spawar.navy.mil, or via the following email address: nesi@spawar.navy.mil.

P1288: Part 2: Traceability

Part 2: Traceability provides a mapping of specific NESI Guidance to other, often more general, high-level DoD net-centric and interoperability efforts such as the Assistant Secretary of Defense for Networks and Information Integration/ Department of Defense Chief Information Officer, or ASD(NII)/DoD CIO, *Net-Centric Checklist*.[R1177] Part 2 includes Perspectives that follow the structure of each high-level effort and provide a NESI interpretation of the implementation implications for program managers and developers which these other efforts direct or imply. These Perspectives, and the associated NESI Guidance and Best Practice links, provide a means of navigating NESI content based on the traceability Part 2 provides. The efforts to which Part 2 content traces may be DoD- or Service-specific; Part 2 currently traces to the following.

Detailed Perspectives

ASD(NII) Net-Centric Guidance [P1239]

Open Technology Development [P1307]

Naval Open Architecture [P1279]

Relationship with the JCIDS Process [P1122]

DISR Service Areas [P1362]

Exposure Verification Tracking Sheets [P1374]

P1239: ASD(NII): Net-Centric Guidance

The ***ASD(NII) Checklist Guidance*** is primarily for managers of new programs or programs that are undergoing a transformation or major upgrade and is especially useful in the pre-systems acquisition and systems acquisition phases. The ASD(NII) Net-Centric Checklist [R1177] uses net-centric design precepts called ***tenets*** to guide the move into the net-centric environment. The design tenets help the DoD leadership understand how net-centricity is evolving. NESI provides specific technical direction for satisfying the Net-Centric Checklist. Note that some tenets address doctrinal or procedural requirements; NESI guidance does not address those areas.

Intended Audience

The Net-Centric Guidance is primarily applicable for new programs or programs that are undergoing a transformation or major upgrade, especially in the pre-systems acquisition and systems acquisition phases. The intended audience for this document includes the following:

- Program managers
- Deputy program managers
- Contracting officers
- Chief engineers
- Contractor personnel
- Enterprise and software architects

Detailed Perspectives

The following perspectives address the ASD(NII) Net-Centric Checklist design tenet categories.

Data [P1244]

Services [P1249]

Information Assurance/Security [P1240]

Transport [P1241]

Each design tenet provides specific technical guidance to enable the system to satisfy its net-centric requirements.

The technical guidance in Part 2 is not necessarily all encompassing; rather, use these guidance statements as part of the overall system engineering analysis of a program to facilitate the evolution of a program or project to net-centricity. Additionally, not all design tenets can be satisfied strictly by technical guidance. All elements of **Doctrine, Organization, Training, Materiel, Leadership, Personnel, and Facilities** (**DOTMLPF**) must participate in the evolution of net-centricity.

P1244: Data

The DoD Net-Centric Data Strategy [R1172] is a key enabler of DoD transformation. Significant attributes of the data strategy include the following:

- Ensuring that data are understandable and trustable, and that they are visible and accessible when and where needed to accelerate decision-making.
- "Tagging" data (intelligence, non-intelligence, raw, and processed) with metadata that supports discovery by both known and unanticipated users in the enterprise.
- Posting data to shared spaces that all users can access, except when limited by security, policy, or regulations.
- Posting in parallel with processing; Task/Post/Process/Use replaces the Task/Process/Exploit/Disseminate paradigm.
- Separating data from applications so that users may choose different applications to exploit the same data.
- Handling information only once to eliminate duplicate, non-authoritative data.

Note: *This section explains the design tenets surrounding data and data assets. A data asset is any entity that involves data. For example, a database is a data asset composed of data records.*

Detailed Perspectives

Design Tenet: Make Data Visible [P1250]
Design Tenet: Make Data Accessible [P1252]
Design Tenet: Make Data Understandable [P1253]
Design Tenet: Make Data Trustable [P1254]
Design Tenet: Make Data Interoperable [P1256]
Design Tenet: Provide Data Management [P1257]
Design Tenet: Be Responsive to User Needs [P1258]

P1250: Design Tenet: Make Data Visible

Data visibility requires an integrated environment of metadata models about the data assets. A data asset is visible when discovery metadata that describes the asset is accessible. Perform forward and/or reverse engineering to capture metadata that describes the data assets of a **node**. Making data visible (even if not accessible) helps develop information about the node and its applications through insights such as the following:

- Essential missions that define the reason for the enterprise; the ultimate goals and objectives that measure enterprise accomplishment
- Procedures performed by various groups in the enterprise that achieve these essential missions
- The specific databases, information systems, and processes that groups use to accomplish aspects of the essential missions
- Context-independent semantic templates of data elements and mechanisms for configuring into data models, as determined by subject matter experts
- Mechanisms for configuring data models into databases used by organizations in the enterprise

Considerations

- Make all data assets visible, even if they are not accessible.
- Use the DoD Discovery Metadata Specification (DDMS) [R1225] and all of its attributes to describe data assets.
- If possible, generate discovery metadata automatically.

Guidance

- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.
- BP1863: Make shareable data assets visible, even if they are not accessible.
- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.

P1252: Design Tenet: Make Data Accessible

Data accessibility requires defining data assets that exist within acceptable boundaries of security, along with the information necessary to access them. **Relational databases** automatically contain metadata about data assets. This perspective extends that definition to **XML** data that may exist independently or that are mapped to and/or from relational data. The following considerations focus on using XML; however, there are alternatives (see the final two Considerations).

XML Requirement

- Use XML to exchange information across systems. Define and implement an XML version of each external interface in all systems. If a system makes data available to external partners, make that data available in the form of an XML document. This is required even if none of the current known partners want or send XML data. Systems may implement other external data exchange mechanisms if an XML interface is supported. Systems may implement other external data exchange mechanisms in addition to an XML interface.

XML Interface Specification

- The system that defines an XML interface will do the following:
  - Specify the syntax of the XML documents it accepts and produces
  - Use the XML Schema standard to express these specifications.
  - Enter the schema in the *DoD Metadata Registry and Clearinghouse*. [R1227] This should occur as early as possible in the development process. Consult designated DoD XML Namespace Managers for guidance in choosing element, attribute, and type identifiers
- An XML interface is responsible for the following actions:
  - Accept input data, producing output data, or both
  - Encode this data in XML documents
  - Specify the schema of the XML documents it accepts and produces
  - Provide documentation that allows programmers and users to understand the meaning of those documents
  - Be implemented by a runtime service that accepts and produces such documents

XML Interface Usage

- A system that uses an XML interface defined by some other system shall record this fact in the DoD Metadata Registry and Clearinghouse.

XML Transport

- Systems must implement one version of each XML interface that is accessible through a **URL** using **HTTP**/**HTTPS**. Systems may implement other versions of the interface using other transport mechanisms, such as **FTP** or **SMTP**, as long as they also support the HTTP version.

Open-Standard Alternatives to XML Format

- Information that is customarily exchanged using a well-known open-standard format does not have to be made available in XML. For example, systems may transfer image data in Joint Photographic Experts Group (JPEG) format, and email messages may continue to use RFC 822 (***Standards for ARPA Internet Text Messages***) headers. It is not necessary to develop an equivalent XML interface for these. Make a list of the exception formats available. It is not necessary to convert information intended for presentation that is currently held in Standard Generalized Markup Language (SGML) format immediately into XML. However, systems should consider future migration from SGML to XML.

Proprietary Alternatives to XML Format

- Information that can only be expressed using closed proprietary formats does not have to be made available in XML. For example, systems may continue to exchange word processor files in Microsoft® Word (DOC format); it is not necessary to develop an equivalent XML interface for this information.

Guidance

- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1763: Indicate the security classification for all classified data.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.

P1253: Design Tenet: Make Data Understandable

Use well-defined standard data elements to establish the semantic basis for data models. To enable data understanding, start with well-defined data ontologies, taxonomies, and vocabularies using standard data elements as the basis for data model structure templates used throughout database models and operating databases. The use of standard data elements also extends to the semantics of **XML schemas** that may exist independently or that are generated from database data models.

Considerations

XML Schema Usage

- Search the **DoD Metadata Registry** for existing XML schemas suitable for reuse in system interfaces. Record the reuse of XML schemas in the DoD Metadata Registry and Clearinghouse.
- If an existing XML schema is close to but not exactly what was specified, review the system requirements with relevant **Communities of Interest** (**COIs**) to determine if the existing schema can be applied as-is or with minor modification.
- Review proposed XML definitions with the designated DoD XML Namespace Manager for relevant COIs.
- Define XML schemas only for that information for which the system is an authoritative source.
- Review XML definitions produced by government and industry consortia for possible reuse.
- Define XML interfaces in collaboration with known information exchange partners.

XML Schema Documentation

- Document the semantics of XML interfaces as annotations on the XML schema.
- Supply a text definition for every element, attribute, and enumeration value defined in the schema. Refer to the ***XML Schema*** specification [R1229] for more information on schema annotations.
- Describe the metadata for each **XML element** with information from related view, physical, logical, conceptual, and data element models.

Guidance

- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1724: Develop XML documents to be well formed.
- G1725: Develop XML documents to be **valid** XML.
- G1726: Define XML Schemas using **XML Schema Definition** (XSD).
- G1727: Provide names for XML type definitions.
- G1728: Define types for all **XML elements**.
- G1729: Annotate XML type definitions.

- G1737: Define a target namespace in schemas.
- G1738: Define a qualified namespace for the target namespace.
- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1759: Use a style guide when developing Web portlets.
- G1761: Provide units of measurements when displaying data.
- G1762: Indicate all simulated data as simulated.
- G1763: Indicate the security classification for all classified data.
- G1770: Explicitly define **Data Distribution Service** (DDS) **Domains**.
- G1796: Explicitly define **Data Distribution Service** (DDS) **Domain Topics**.
- G1798: Explicitly define all the **Data Distribution Service** (DDS) **Domain data types**.
- G1799: Explicitly associate data types to the **Data Distribution Service** (DDS) **Topics** within a DDS **Domain**
- G1800: Explicitly identify Keys within the **Data Distribution Service** (DDS) **data type** that uniquely identify an instance of a data object.
- G1810: Use **data models** to document the data contained within the **Data Distribution Service** (DDS) **Data-Centric Publish Subscribe** (DCPS).

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.

P1254: Design Tenet: Make Data Trustable

A key to supporting data trust relationships is to ensure that data is unchanged (or otherwise reconcilable) when the data is accessed from all points within the trust relationship. Formalize and enforce authoritative data sources and ensure that the data is current and distributed in a timely manner.

Considerations

- Use the Resource Descriptors and Security Descriptors specified by the **DoD Metadata Registry** to provide data validity and security information.
- Identify the authoritative source and purpose for each **data element**.
- Aggregated data can often exceed the security level of the individual data elements. Recognize and account for the possibility of an increased security level when aggregating data.

Guidance

- G1154: Use **stored procedures** for operations that are focused on the insertion and maintenance of data.
- G1155: Use **triggers** to enforce **referential** or data integrity, not to perform complex **business logic**.
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1762: Indicate all simulated data as simulated.
- G1763: Indicate the security classification for all classified data.

P1256: Design Tenet: Make Data Interoperable

To be interoperable, data must have known structural and discovery metadata as well as mechanisms to support its translation (e.g., to different units). Analyze and register metadata data assets such as names, data types, lengths, precision, scale, and restricted value domains. Identify the standards used to represent these items. Work with **Communities of Interest** to ensure the data represents appropriate semantics.

Considerations

XML Wrapped Data

- If XML wrapped data are intended for exchange, configure them in terms of standard transactions with headers, trailers, and bodies.

XML Schema Validation

- Systems that produce XML documents shall guarantee that the XML documents are valid according to the XML schema they have published in the **DoD Metadata Registry**. Systems that receive XML documents should validate them against the schemas published by the Source system.

Guidance

- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1724: Develop XML documents to be well formed.
- G1725: Develop XML documents to be **valid** XML.
- G1726: Define XML Schemas using **XML Schema Definition** (XSD).
- G1729: Annotate XML type definitions.
- G1737: Define a target namespace in schemas.
- G1738: Define a qualified namespace for the target namespace.
- G1746: Develop XSLT **style sheets** that are XSLT version agnostic.
- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1754: Give each new XML schema version a unique URL.
- G1759: Use a style guide when developing Web portlets.
- G1761: Provide units of measurements when displaying data.
- G1763: Indicate the security classification for all classified data.
- G1770: Explicitly define **Data Distribution Service** (DDS) **Domains**.
- G1772: Assign a unique identifier for each **Data-Distribution Service** (DDS) **Domain**.
- G1798: Explicitly define all the **Data Distribution Service** (DDS) **Domain data types**.
- G1799: Explicitly associate data types to the **Data Distribution Service** (DDS) **Topics** within a DDS **Domain**

- G1800: Explicitly identify Keys within the **Data Distribution Service** (DDS) **data type** that uniquely identify an instance of a data object.
- G1810: Use **data models** to document the data contained within the **Data Distribution Service** (DDS) **Data-Centric Publish Subscribe** (DCPS).
- G1796: Explicitly define **Data Distribution Service** (DDS) **Domain Topics**.
- G1001: Use formal standards to define public **interfaces**.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.
- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.
- BP1866: Coordinate with end users to develop interoperable materiel in support of high-value mission capability.

P1257: Design Tenet: Provide Data Management

Enhance the ability to support data management by providing a process to define, develop, and maintain an ontology (e.g., schemas, thesauruses, vocabularies, keyword lists, and taxonomies).

Considerations

- Obtain metrics to promote awareness of data management successes and areas requiring improvement.
- Provide a graphical representation, outline, or model representing the format, structure, and relationship of data.

Guidance

- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1726: Define XML Schemas using **XML Schema Definition** (XSD).
- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1729: Annotate XML type definitions.
- G1647: Provide access to the **Federated Search** Services.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.
- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.

Examples

- A database table and relationship structure
- A **document type definition** (**DTD**)
- A data structure used to pass information between systems
- An XML schema document (XSD) that represents a data structure and related information encoded as XML

P1258: Design Tenet: Be Responsive to User Needs

Include users in processes for creating discoverable, accessible, understandable, and trusted information and services. Understanding information interoperability creates an environment that can be responsive to users. User feedback mechanisms provide a means of capturing and reporting user satisfaction and give portfolio managers decision making information to steer investments, developments and improvements. Service and information providers in a mission area should work together to define the processes for using the user feedback for service and information improvements because these processes are specific to a portfolio of capabilities in the enterprise.

Considerations

- Provide a capability for capturing, tracking, and responding to user feedback.
- Collaborate with **Communities of Interest** (**COIs**) in responding to user feedback.
- Ensure that user feedback is visible to the net-centric environment.
- Ensure that processes exist for consumers to do the following:
  - Request additional information from the information provider
  - Request changes in the format, i.e., syntax or semantics, of visible information
  - Report a problem with the information
- Establish metrics for determining responsiveness to user needs.

Guidance

- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1571: Maintain a comprehensive list of all the **Communities of Interest** (**COIs**) to which the **Components** of a Node belong.
- G1575: Designate Node representatives to relevant **Communities of Interest** (**COIs**) in which Components of the Node participate.
- G1760: Solicit feedback from users on user interface usability problems.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.
- BP1867: Use metrics to track responsiveness to user information sharing needs.

P1249: Services

A service is a contractually defined behavior a software component provides through a well-defined, published and shareable interface. The service concept is based on implementation characteristics like loose coupling, location independence, etc., that are inherently **net-centric**; this enables the rapid development and deployment of capabilities that, combined with other services, can provide a range of simple and complex functions that could be shared across diverse applications and management boundaries and woven into mission threads or business flows.

> ***Note:*** *For more information on service characteristics see the Service-Oriented Architecture [P1304] perspective in Part 1.*

Detailed Perspectives

Design Tenet: Service-Oriented Architecture (SOA) [P1259]
Design Tenet: Open Architecture [P1268]
Design Tenet: Scalability [P1270]
Design Tenet: Availability [P1271]
Design Tenet: Accommodate Heterogeneity [P1275]
Design Tenet: Decentralized Operations and Management [P1276]
Design Tenet: Enterprise Service Management [P1278]

P1259: Design Tenet: Service-Oriented Architecture (SOA)

Service-Oriented Architecture (SOA) is an architectural design style for building flexible, adaptable and distributed computing environments where functionality is exposed and shared across enterprise by the means of services.

> ***Note:*** *For more information on service-oriented architecture and service characteristics that enable the sharing of services across an enterprise see the Service-Oriented Architecture [P1304] perspective in Part 1.*

Web Services

- Build **Web services** in accordance with the technical standards and conformance requirements prescribed by the current version of the WS-I Basic Profile.[R1237]
  - Use the WS-I Sample Application as a model for implementing and documenting Web services.
  - Use test tools authorized by WS-I that verify conformance with the current version of the WS-I Basic Profile.
  - Build and develop security extensions as prescribed in the current version of the WS-I Basic Security Profile.

Service Description

- Describe services using a standard Service Definition Framework (SDF). The Service Definition Framework [P1296] perspective provides a detailed specification for service definition and implementation. The SDF should address the following information for each service:
  - What the service does
  - How the service works (from a "black box" perspective)
  - Required security mechanisms or restrictions
  - Performance or quality of service (Q0S) information
  - Points of contact for the service
  - The specifics of how to bind to (access or use) the service

Service Access Point (SAP)

- Describe services provided by a system's SAPs. From a service provider perspective, SAPs can be abstracted away from the back-end or internal processing activities of the service. Looser coupling between SAP and service internals enables a service provider to change the internal workings of the back end, such as moving to a new version of a database, without changing the SAP.

Service Design

- Design services around operational requirements and service consumers' needs.
  - Base the service specifications on the needs of the initial users, since it is impossible to know all the possible service consumers.
  - Provide an extensible interface so the service design can support future needs.

Service Design Characteristics

- Design services in accordance with best practices and patterns. For example, a service design should specify the information objects that are communicated across its interface in terms of enterprise metadata (e.g., time, location). These enable semantic agreement between the information objects.
- Design information objects to minimize the number of transactions across the service interface. An example of this is a request for an Authority to Operate (ATO), possibly constrained by a time and location attribute, followed by a reply containing the ATO that is applicable to a specific area of interest and time.

Service Implementation Characteristics

- Implementation information focuses on the technical implementation details that prospective service developers or providers need to design new services, or a service that uses another service. These attributes typically include items like the **WSDL** description of the service, details of a service's **API** interface point, and a description of service dependencies. Implement services using the following practices:
  - Document the open standards used.
  - Use vendor and platform independent messages.
  - Identify addresses using **Uniform Resource Identifiers** (**URIs**).
  - Use defined and documented service interfaces.
  - Register XML interface descriptions using the **DoD Metadata Registry**.
  - Pass enterprise or COI objects, defined by their respective metadata, across its service interface.
  - Use extensible service interfaces with versioning, independent of the interface implementation version.

Service Level Agreement (SLA)

- Document a **Service Level Agreement** to do the following:
  - Include quantitative measures for service usage, performance analysis, continuity of operations plan, and performance across the range of bandwidths provided by the node.
  - Have terms that the node's management services can monitor and manage.
  - Define responsibility for day-to-day service operations and procedures for reporting problems.

Service Interfaces

- Interface information should include descriptions of service features, service functionality, service provider identification, instructions on how to access and use the service through the SAP, and so on. The interface information should also discuss the different form factors that a service supports, such as a PDA.
- Express the Web service interfaces in WSDL in accordance with the current version of the WS-I Basic Profile.
- Register all XML schema files imported into WSDL under the appropriate namespace in the **DoD XML Registry**.
- At a minimum, store WSDL files in a file accessible via **URL** and **HTTP**.

Node Responsibilities for Services

- The node infrastructure should enable mission application software to be instantiated as services; this includes software libraries that support **SOAP** and WSDL processing. Node responsibilities include the following:
  - Using Web services standards (SOAP and WSDL) to interoperate applications across nodes.
  - Providing secure access to components in accordance with node and **GIG** IA/Security policies and services.
  - Designing services to be managed by the node in accordance with enterprise policy. Management services will typically be part of the node component framework environment (e.g., **Java EE** application server, .NET management environment) that is used in conjunction with NCES Enterprise Service Management.
  - Providing the capability to name and register components for local use within the node (e.g., **JNDI**). Component registration mechanisms shall interface or extend to service registration mechanisms, such as registration in the NCES Discovery service. If the component is only visible to the local node, it does not have to be registered in the NCES Discovery service.

Service Registration

- Systems register services using the standard service metadata in a directory available to the nodes in the enterprise. This directory may be based in the node, in an NCES Discovery Service, or both. At a minimum, identify a service by a Uniform Resource Identifier.
- Nodes register services as resources with the NCES Policy Management Service and control access to services using the NCES Policy Decision Services. The NCES Resource Attribute Services must provide access to service attributes.

Service Security

- Security information provides detailed information about the security specifications of the service, such as restrictions on who can use or access the service, for example indicating that the user must present a valid DoD PKI certificate to access the service.
- A security framework is required at the node level to authenticate principals, ensure confidentiality and integrity of messages and authorize access.
- Use security mechanisms provided by the node. These must include mutual authentication over an encrypted channel such as SSL, authorization, confidentiality, integrity and non-repudiation.
- Services must support **role-based access control** (**RBAC**) mechanisms.
- Nodes should provide interfaces to NCES security services.
- Nodes should establish trust relationships with other nodes in the enterprise using the NCES Domain Federation Service.

Support for Service Orchestration

- Provide the capability to compose mission capabilities from one or more services using a service orchestration or workflow mechanism based on industry standards such as **BPEL**.

Guidance

- G1001: Use formal standards to define public **interfaces**.
- G1002: Separate public **interfaces** from implementation.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1004: Make public **interfaces** backward-compatible within the constraints of a published **deprecation** policy.
- G1008: Isolate platform-specific **interfaces.**
- G1010: Use **open-standard** logging frameworks.
- G1011: Make components independently deployable.
- G1012: Use a set of services to expose **Component** functionality.
- G1014: Access databases through **open standard** interfaces.
- G1018: Assign version identifiers to all public interfaces.
- G1019: Deprecate public interfaces in accordance with a published deprecation policy.
- G1022: Insulate public **interfaces** from compile-time dependencies.
- G1027: Internally document all source code developed with DoD funding.
- G1030: Use a user interface **component** library.
- G1032: Validate all input fields.
- G1035: Follow W3C standards for code which will generate a Web page display.
- G1043: Separate formatting from data through the use of **style sheets** instead of hard coded **HTML** attributes.
- G1044: Comply with Federal accessibility standards contained in Section 508 of the Rehabilitation Act of 1973 (as amended) when developing software user interfaces.
- G1045: Define **XML** format information separately in **XSL**.
- G1050: In **ASP**, isolate the presentation tier from the middle tier using **COM** objects.
- G1052: Use the code-behind feature in ASP.NET to separate presentation code from the business logic.
- G1053: Do not embed HTML code in any code-behind code used by aspx pages.
- G1056: Specify a versioning policy for **.NET** assemblies.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.
- G1060: Encapsulate Java code that is used in **JSP**(s) in tag libraries.
- G1071: Use vendor-neutral interface connections to the enterprise (e.g., **LDAP**, **JNDI**, **JMS**, databases).

- G1073: Isolate vendor extensions to enterprise-services standard interfaces.
- G1078: Document the use of non-**Java EE**-defined **deployment descriptors**.
- G1079: Isolate tailorable data values into the **deployment descriptors** for **Java EE** applications.
- G1080: Adhere to the **Web Services Interoperability Organization** (**WS-I**) Basic Profile specification for **Web service** environments.
- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).
- G1083: Do not pass **Web Services-Interoperability Organization** (**WS-I**) **Document Object Model** (**DOM**) documents as strings.
- G1085: Establish a **registered namespace** in the **XML Gallery** in the **DoD Metadata Registry** for all DoD Programs.
- G1087: Validate all **Web Services Definition Language** (**WSDL**) files that describe **Web services**.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1090: Do not **hard-code** a **Web service's endpoint**.
- G1093: Implement exception handlers for **SOAP**-based **Web services**.
- G1095: Use **W3C** fault codes for all **SOAP** faults.
- G1118: Localize **CORBA** vendor-specific source code into separate **modules**.
- G1119: Isolate user-modifiable configuration parameters from the **CORBA** application source code.
- G1121: Do not modify **CORBA** Interface Definition Language (**IDL**) compiler auto-generated stubs and skeletons.
- G1123: Use the Fat Operation Technique in **IDL** operator invocation.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.
- G1127: Use a **UDDI** specification that supports publishing discovery services.
- G1131: Use industry standard Universal Description, Discovery, and Integration (**UDDI**) **APIs** for all UDDI inquiries.
- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.
- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1144: Develop two-level database models: one level captures the **conceptual** or logical aspects, and the other level captures the **physical** aspects.
- G1146: Include information in the **data model** necessary to generate a **data dictionary**.
- G1147: Use **domain analysis** to define the constraints on input data validation.
- G1148: **Normalize** data models.
- G1151: Define declarative **foreign keys** for all relationships between tables to enforce **referential integrity**.
- G1153: Separate application, presentation, and data tiers.
- G1154: Use **stored procedures** for operations that are focused on the insertion and maintenance of data.
- G1155: Use **triggers** to enforce **referential** or data integrity, not to perform complex **business logic**.
- G1190: Use a build tool.
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.
- G1204: Create configuration services to provide distributed user control of the appropriate configuration parameters.
- G1205: Use non-source code persistence to store all user-modifiable **CORBA** service configuration parameters.
- G1208: Add new functionality rather than redefining existing interfaces in a manner that brings incompatibility.
- G1209: For Java, use **JDK** logging facilities.
- G1210: For **.NET**, use Debug and Trace from the **`System.Diagnostics` namespace**.

- G1217: Develop and use externally configurable components.
- G1218: Use a build tool that supports operation in an automated mode.
- G1219: Use a build tool that checks out files from configuration control.
- G1220: Use a build tool that **compiles** source code and dependencies that have been modified.
- G1221: Use a build tool that creates libraries or archives after all required compilations are completed.
- G1222: Use a build tool that creates executables.
- G1223: Use a build tool that is capable of running unit tests.
- G1224: Use a build tool that cleans out intermediate files that can be regenerated.
- G1225: Use a build tool that is independent of the **Integrated Development Environment**.
- G1237: Do not **hard-code** the configuration data of a **Web service** vendor.
- G1239: Use **design patterns** (e.g., **facade**, **proxy**, or **adapter**) or property files to isolate vendor-specifics of vendor-dependent connections to the enterprise.
- G1245: Isolate the **Web service portlet** from platform dependencies using the **Web Services for Remote Portlets** (**WSRP**) Specification protocol.
- G1267: Use industry standard **HTML** data entry fields on **Web pages**.
- G1268: Label all data entry fields.
- G1270: Include scroll bars for text entry areas if the data buffer is greater than the viewable area.
- G1271: Provide instructions and **HTML** examples for all style sheets.
- G1276: Do not modify the contents of the Web browser's status bar.
- G1277: Do not use tickers on a Web site.
- G1278: Use the browser default setting for links.
- G1283: Use **linked style sheets** rather than embedded styles.
- G1284: Use only one font for **HTML** body text.
- G1285: Use **relative font sizes**.
- G1286: Provide text labels for all buttons.
- G1287: Provide feedback when a transaction will require the user to wait.
- G1292: Use text-based Web site navigation.
- G1294: Provide a site map on all Web sites.
- G1295: Provide redundant text links for images within an **HTML** page.
- G1356: Use the **SOAP** standard for all **Web services**.
- G1566: Use **`alt`** attributes to provide alternate text for non-text items such as images.
- G1569: Maintain a comprehensive list of all of the **Components** that are part of the Node.
- G1573: Define the enterprise design patterns that a Node supports.
- G1574: Define which enterprise design patterns a **Component** requires.
- G1579: Define which **Enterprise Services** the Node will host locally when the Node becomes operational.
- G1580: Define which **Enterprise Services** will be hosted over the **Global Information Grid** (GIG) when the Node becomes operational.
- G1581: Expose legacy functionality through the use of a service.
- G1635: Make Nodes that will be part of the **Global Information Grid** (**GIG**) consistent with the *GIG Integrated Architecture*.
- G1636: Comply with the **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**).
- G1637: Make Node-implemented **directory services** comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**).

- G1638: Comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node directory services **proxies**.
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.
- G1641: Comply with the Service Discovery **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node-implemented **Service Discovery** (**SD**).
- G1642: Comply with the **Service Discovery Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node **Service Discovery** (**SD**) **proxies**.

Best Practices

- BP1007: Develop software using **open standard Application Programming Interfaces** (**APIs**).
- BP1021: Create fully encapsulated classes.
- BP1863: Make shareable data assets visible, even if they are not accessible.

P1268: Design Tenet: Open Architecture

Design mission application software to be separable from the supporting **node** and to access the node through public interfaces based on standards governed by a recognized standards organization (e.g., IEEE, W3C, OASIS).

Component Based

- Architect mission application software in the node as components integrated within a node. Provide run-time and resource management services (e.g., component management, security, virtual machines, memory management, object management, resource pooling).
- Include component frameworks in the node based on commercially available solutions without proprietary extensions. Wrap any extensions, if used, via the appropriate design pattern.
- Architect and manage mission application software that spans multiple nodes in a manner that aligns with all of the supporting nodes.

> ***Note:*** *Examples include* ***Java Platform, Enterprise Edition*** *(****Java EE****),* ***Common Object Request Broker Architecture*** *(****CORBA****),* ***.NET Framework****, and* ***Data Distribution System*** *(****DDS****).*

Public Interfaces

- Provide the mechanism on the node for components to expose public interfaces. The interface must be separate from the implementation. Base the public interface mechanism on the node component framework. These public interfaces must be visible to other components in the node.

Layered Software Architecture

- Layer application software using an N-tier architecture. At a minimum, use discrete ***client***, ***presentation***, ***middle***, and ***data*** tiers.
  - ***Client Tier*** -The client tier supports a wide range of device types such as desktop computers, laptops, mobile, wireless, and personal digital assistant (PDA). It supports direct interaction with the user.
  - ***Presentation Tier -*** The presentation tier provides content to a range of client device types supported by the node (e.g., Hypertext, eXtensible or Wireless Markup Language [**HTML**, **XML**, **WML**]). Implement presentation components with the mechanisms in the node's component framework.
  - ***Middle Tier -*** The middle tier supports the construction of componentized business logic and public interfaces (e.g., interface classes). Base business components on programming mechanisms provided by the component framework chosen by the node (e.g. **Enterprise Java Beans**, CORBA services, **COM** components). Specific business logic elements, such as data validation, may reside in other tiers.
  - ***Data Tier -*** Base access to the data tier within nodes on industry open-standard mechanisms such a **SQL** or **JDBC**/**ODBC**. Use services to access data across nodes.

Wrapping Legacy Systems

- Wrap legacy application software with an interface that is accessible from the node; for example, use Java Connector Architecture on a Java EE platform. See (e.g., Pattern: Wrapping Legacy Code into a Service [P1219]) for additional information on wrapping legacy systems.

Guidance

- G1001: Use formal standards to define public **interfaces**.
- G1002: Separate public **interfaces** from implementation.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1004: Make public **interfaces** backward-compatible within the constraints of a published **deprecation** policy.
- G1008: Isolate platform-specific **interfaces.**
- G1010: Use **open-standard** logging frameworks.
- G1011: Make components independently deployable.

- G1012: Use a set of services to expose **Component** functionality.
- G1014: Access databases through **open standard** interfaces.
- G1018: Assign version identifiers to all public interfaces.
- G1019: Deprecate public interfaces in accordance with a published deprecation policy.
- G1022: Insulate public **interfaces** from compile-time dependencies.
- G1027: Internally document all source code developed with DoD funding.
- G1030: Use a user interface **component** library.
- G1032: Validate all input fields.
- G1035: Follow W3C standards for code which will generate a Web page display.
- G1043: Separate formatting from data through the use of **style sheets** instead of hard coded **HTML** attributes.
- G1044: Comply with Federal accessibility standards contained in Section 508 of the Rehabilitation Act of 1973 (as amended) when developing software user interfaces.
- G1045: Define **XML** format information separately in **XSL**.
- G1050: In **ASP**, isolate the presentation tier from the middle tier using **COM** objects.
- G1052: Use the code-behind feature in ASP.NET to separate presentation code from the business logic.
- G1053: Do not embed HTML code in any code-behind code used by aspx pages.
- G1056: Specify a versioning policy for **.NET** assemblies.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.
- G1060: Encapsulate Java code that is used in **JSP**(s) in tag libraries.
- G1071: Use vendor-neutral interface connections to the enterprise (e.g., **LDAP**, **JNDI**, **JMS**, databases).
- G1073: Isolate vendor extensions to enterprise-services standard interfaces.
- G1078: Document the use of non-**Java EE**-defined **deployment descriptors**.
- G1079: Isolate tailorable data values into the **deployment descriptors** for **Java EE** applications.
- G1080: Adhere to the **Web Services Interoperability Organization** (**WS-I**) Basic Profile specification for **Web service** environments.
- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).
- G1083: Do not pass **Web Services-Interoperability Organization** (**WS-I**) **Document Object Model** (**DOM**) documents as strings.
- G1085: Establish a **registered namespace** in the **XML Gallery** in the **DoD Metadata Registry** for all DoD Programs.
- G1087: Validate all **Web Services Definition Language** (**WSDL**) files that describe **Web services**.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1090: Do not **hard-code** a **Web service's endpoint**.
- G1093: Implement exception handlers for **SOAP**-based **Web services**.
- G1095: Use **W3C** fault codes for all **SOAP** faults.
- G1118: Localize **CORBA** vendor-specific source code into separate **modules**.
- G1119: Isolate user-modifiable configuration parameters from the **CORBA** application source code.
- G1121: Do not modify **CORBA** Interface Definition Language (**IDL**) compiler auto-generated stubs and skeletons.
- G1123: Use the Fat Operation Technique in **IDL** operator invocation.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.
- G1127: Use a **UDDI** specification that supports publishing discovery services.
- G1131: Use industry standard Universal Description, Discovery, and Integration (**UDDI**) **APIs** for all UDDI inquiries.

- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.
- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1144: Develop two-level database models: one level captures the **conceptual** or logical aspects, and the other level captures the **physical** aspects.
- G1153: Separate application, presentation, and data tiers.
- G1190: Use a build tool.
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.
- G1204: Create configuration services to provide distributed user control of the appropriate configuration parameters.
- G1205: Use non-source code persistence to store all user-modifiable **CORBA** service configuration parameters.
- G1208: Add new functionality rather than redefining existing interfaces in a manner that brings incompatibility.
- G1209: For Java, use **JDK** logging facilities.
- G1210: For **.NET**, use Debug and Trace from the **`System.Diagnostics`** **namespace**.
- G1213: Provide an architecture design document.
- G1214: Provide a document with a plan for **deprecating** obsolete **interfaces**.
- G1215: Provide a coding standards document.
- G1216: Provide a software release plan document.
- G1217: Develop and use externally configurable components.
- G1218: Use a build tool that supports operation in an automated mode.
- G1219: Use a build tool that checks out files from configuration control.
- G1220: Use a build tool that **compiles** source code and dependencies that have been modified.
- G1221: Use a build tool that creates libraries or archives after all required compilations are completed.
- G1222: Use a build tool that creates executables.
- G1223: Use a build tool that is capable of running unit tests.
- G1224: Use a build tool that cleans out intermediate files that can be regenerated.
- G1225: Use a build tool that is independent of the **Integrated Development Environment**.
- G1237: Do not **hard-code** the configuration data of a **Web service** vendor.
- G1239: Use **design patterns** (e.g., **facade**, **proxy**, or **adapter**) or property files to isolate vendor-specifics of vendor-dependent connections to the enterprise.
- G1245: Isolate the **Web service portlet** from platform dependencies using the **Web Services for Remote Portlets** (**WSRP**) Specification protocol.
- G1267: Use industry standard **HTML** data entry fields on **Web pages**.
- G1271: Provide instructions and **HTML** examples for all style sheets.
- G1276: Do not modify the contents of the Web browser's status bar.
- G1278: Use the browser default setting for links.
- G1284: Use only one font for **HTML** body text.
- G1285: Use **relative font sizes**.
- G1356: Use the **SOAP** standard for all **Web services**.
- G1573: Define the enterprise design patterns that a Node supports.
- G1574: Define which enterprise design patterns a **Component** requires.
- G1581: Expose legacy functionality through the use of a service.
- G1626: Identify which **Core Enterprise Services** (**CES**) capabilities the Node **Components** require.

- G1627: Identify the priority of each **Core Enterprise Services** (**CES**) capability the Node **components** require.
- G1629: Identify which **Net-Centric Enterprise Services** (**NCES**) capabilities the Node requires during deployment.
- G1630: Comply with the applicable **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) for implemented **Core Enterprise Services** (**CES**) in the Node.
- G1631: Expose **Core Enterprise Services** (**CES**) that comply with the applicable **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in all Node services **proxies**.
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.
- G1724: Develop XML documents to be well formed.
- G1725: Develop XML documents to be **valid** XML.
- G1726: Define XML Schemas using **XML Schema Definition** (XSD).
- G1727: Provide names for XML type definitions.
- G1728: Define types for all **XML elements**.
- G1729: Annotate XML type definitions.
- G1737: Define a target namespace in schemas.
- G1738: Define a qualified namespace for the target namespace.
- G1746: Develop XSLT **style sheets** that are XSLT version agnostic.
- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1754: Give each new XML schema version a unique URL.
- G1770: Explicitly define **Data Distribution Service** (DDS) **Domains**.

Best Practices

- BP1007: Develop software using **open standard Application Programming Interfaces** (**APIs**).
- BP1021: Create fully encapsulated classes.
- BP1863: Make shareable data assets visible, even if they are not accessible.
- BP1864: Layer architectures to support clear boundaries between data management, presentation, and business logic functionality.

P1270: Design Tenet: Scalability

Design services and components to use resource management mechanisms that the hosting Node provides to enable scalability under load. For example, use buffer and connection pools, tuned to the expected user load, to enable concurrent user sessions with acceptable performance.

- Scalability is the extent to which the organization, program, project, or initiative can grow to accommodate additional users. Scalable components are either co-located or globally distributed. Scalability of computing infrastructure (CI) components and CI-related **doctrine, organization, training, materiel, leadership and education, personnel, and facilities** (**DOTMLPF**) allows for rapidly implemented increases in capacity and capability to support program, project, and initiative growth or dynamically changing requirements.

To the greatest extent possible given bandwidth and technical environment considerations, make services accessible in an open-systems, interface-driven, distributed computing environment with reusable components available to the enterprise. Acceptable Web-based methods are represented by Internet standards and protocols registered in the **Defense IT Standards Registry** (**DISR**) and managed by the DoD IT Standards Committee (ITSC). To the greatest extent possible, the service design should include considerations for potential edge users with limited bandwidth access and limited display or storage capacity. As enterprise services emerge, the infrastructure should establish new parameters related to maintainability, scalability, performance, orchestration, accreditation, and availability.

Considerations

Design Factors

- System architects, program managers, and designers for a program, project or initiative should consider a vision that includes growth projections for the program's foreseeable future.

Assessing Scalability Requirements

- Assess and evaluate requirements and capabilities of services to understand scalability hot spots better
- Properly estimate usage patterns
- Manage user authentication/authorization
- Manage session state where applicable
- Scale user or internal facing Web sites
- Scale data resources
- Scale CPU load

Stateless Service

- Each message that a consumer sends to a provider must contain all necessary information for the provider to process it. This constraint makes a service provider more scalable because the provider does not have to store state information between requests.

Stateful Service

- Stateful service is difficult to avoid in a number of situations. For example, establishing a session between a consumer and a provider for efficiency reasons such as sending a security certificate with each request. The process creates a load for both consumer and provider. It is much quicker to replace the certificate with a token shared just between the consumer and provider. Stateful services require both the consumer and the provider to share the same consumer-specific context, which is either included in or referenced by messages exchanged between the provider and the consumer. The problem with this constraint is that it potentially reduces the overall scalability of the service. The service provide must remember context for each consumer. Coupling between a service provider and a consumer is increased. Switching service providers is more difficult.

Guidance

- G1012: Use a set of services to expose **Component** functionality.
- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).

- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1123: Use the Fat Operation Technique in **IDL** operator invocation.
- G1153: Separate application, presentation, and data tiers.
- G1283: Use **linked style sheets** rather than embedded styles.
- G1352: Use database clustering and redundant array of independent disks (RAID) for high availability of data.
- G1572: Include the Node as a party to any **Service Level Agreements** (**SLAs**) signed by any of the **components** of the Node.

Best Practices

- BP1864: Layer architectures to support clear boundaries between data management, presentation, and business logic functionality.

P1271: Design Tenet: Availability

As the net-centric environment evolves, an ever increasing number of information services will become available to DoD users. At the same time, infrastructure support for these services will also transform to net-centric standards, leveraging shared processing and storage on the GIG and dynamic allocation. It will be critical in this environment to maintain acceptable and measurable levels of support for all **enterprise** capabilities. When users seek, find and use an **Enterprise Service**, they will have certain expectations regarding its pedigree, reliability and availability. These attributes should be consistent across all Enterprise Services.

Design services and components to meet the availability requirements of the node. The implementation should use the maintenance strategies and management mechanisms provided by the Node's infrastructure.

Considerations

- While an Enterprise Service may be provided from anywhere in the **Global Information Grid** (**GIG**), user expectations demand that they be hosted in environments that meet minimum GIG computing node standards in terms of availability, support and backup.

Guidance

- G1352: Use database clustering and redundant array of independent disks (RAID) for high availability of data.
- G1572: Include the Node as a party to any **Service Level Agreements** (**SLAs**) signed by any of the **components** of the Node.

Best Practices

- BP1868: Incorporate mechanisms to enhance the survivability, resiliency, redundancy, and reliability of Computing Infrastructure (CI).

P1275: Design Tenet: Accommodate Heterogeneity

The **Global Information Grid** (**GIG**) is a heterogeneous environment. No one product will meet the needs of potentially vastly different operational environments. Services and Service-Oriented Architecture (SOA) related infrastructure will need to interoperate across these diverse environments.

Service Structure

- Design systems to be able to deploy services separately from the supporting **node**. The services should access the node through public interfaces.

Service Configuration

- Design systems to be able to configure services on each node on which they are deployed. Use external configuration file mechanisms (e.g., deployment descriptors for **Java EE** applications) to specify the configuration. Do not use hard-coded configuration parameters that require a binary tool to update or that require a recompile and relink.

Node Structure

- Nodes provide the infrastructure and rules for assembling, configuring, deploying, securing, operating, and managing mission applications and services. For more information, see NESI Part 4: Node Guidance [P1130] [P1130 [P1130]].
- Nodes are responsible for provisioning their diverse mission application and services. They must configure and operate them in accordance with enterprise management policy.

Guidance

- G1001: Use formal standards to define public **interfaces**.
- G1002: Separate public **interfaces** from implementation.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1004: Make public **interfaces** backward-compatible within the constraints of a published **deprecation** policy.
- G1008: Isolate platform-specific **interfaces.**
- G1010: Use **open-standard** logging frameworks.
- G1011: Make components independently deployable.
- G1012: Use a set of services to expose **Component** functionality.
- G1014: Access databases through **open standard** interfaces.
- G1018: Assign version identifiers to all public interfaces.
- G1019: Deprecate public interfaces in accordance with a published deprecation policy.
- G1022: Insulate public **interfaces** from compile-time dependencies.
- G1030: Use a user interface **component** library.
- G1032: Validate all input fields.
- G1035: Follow W3C standards for code which will generate a Web page display.
- G1043: Separate formatting from data through the use of **style sheets** instead of hard coded **HTML** attributes.
- G1044: Comply with Federal accessibility standards contained in Section 508 of the Rehabilitation Act of 1973 (as amended) when developing software user interfaces.
- G1045: Define **XML** format information separately in **XSL**.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.
- G1071: Use vendor-neutral interface connections to the enterprise (e.g., **LDAP**, **JNDI**, **JMS**, databases).
- G1073: Isolate vendor extensions to enterprise-services standard interfaces.
- G1080: Adhere to the **Web Services Interoperability Organization** (**WS-I**) Basic Profile specification for **Web service** environments.

- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).
- G1083: Do not pass **Web Services-Interoperability Organization** (**WS-I**) **Document Object Model** (**DOM**) documents as strings.
- G1087: Validate all **Web Services Definition Language** (**WSDL**) files that describe **Web services**.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1090: Do not **hard-code** a **Web service's endpoint**.
- G1093: Implement exception handlers for **SOAP**-based **Web services**.
- G1095: Use **W3C** fault codes for all **SOAP** faults.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.
- G1127: Use a **UDDI** specification that supports publishing discovery services.
- G1131: Use industry standard Universal Description, Discovery, and Integration (**UDDI**) **APIs** for all UDDI inquiries.
- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.
- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1153: Separate application, presentation, and data tiers.
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.
- G1204: Create configuration services to provide distributed user control of the appropriate configuration parameters.
- G1208: Add new functionality rather than redefining existing interfaces in a manner that brings incompatibility.
- G1209: For Java, use **JDK** logging facilities.
- G1210: For **.NET**, use Debug and Trace from the **`System.Diagnostics`** **namespace**.
- G1217: Develop and use externally configurable components.
- G1237: Do not **hard-code** the configuration data of a **Web service** vendor.
- G1239: Use **design patterns** (e.g., **facade**, **proxy**, or **adapter**) or property files to isolate vendor-specifics of vendor-dependent connections to the enterprise.
- G1245: Isolate the **Web service portlet** from platform dependencies using the **Web Services for Remote Portlets** (**WSRP**) Specification protocol.
- G1267: Use industry standard **HTML** data entry fields on **Web pages**.
- G1271: Provide instructions and **HTML** examples for all style sheets.
- G1276: Do not modify the contents of the Web browser's status bar.
- G1278: Use the browser default setting for links.
- G1284: Use only one font for **HTML** body text.
- G1285: Use **relative font sizes**.
- G1292: Use text-based Web site navigation.
- G1295: Provide redundant text links for images within an **HTML** page.
- G1566: Use **`alt`** attributes to provide alternate text for non-text items such as images.
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.

Best Practices

- BP1007: Develop software using **open standard Application Programming Interfaces** (**APIs**).

- BP1021: Create fully encapsulated classes.
- BP1864: Layer architectures to support clear boundaries between data management, presentation, and business logic functionality.

P1276: Design Tenet: Decentralized Operations and Management

Design services to provide a management interface that either the **node's** management services or the **Net-Centric Enterprise Services** (**NCES**) Enterprise Service Management services can access. Intuitive management interfaces provide operators with the toolset to be responsive to system operations, system changes, and maintenance needs. Design management interfaces that new personnel can easily learn with minimum training to mitigate loss of knowledge and skill sets caused by troop rotation or personnel turnover. Use **COTS** products with Web-based **GUIs** that enable operators or administrators to make configuration changes easily, execute maintenance utilities (e.g., log capture, backups), check operational performance/status, and facilitate user administration.

Considerations

- Support a decentralized operational concept where other systems, services, or capabilities are providing key elements of the end-to-end **net-centric** solution.
- Provide an integrated digital environment to enhance communications and productivity for management and operations of programs, projects or initiatives.
- Provide remote management capabilities that are employed to manage the distributed computing infrastructure such as Telnet, Secure Shell, Web-based proprietary, Web-based COTS or customized COTS, or other technologies.
- Provide security and access control mechanisms to facilitate management across differing security domains in the DoD, Intelligence Community, other government agencies, and coalition partners.

Guidance

- G1204: Create configuration services to provide distributed user control of the appropriate configuration parameters.
- G1606: Manage **routers** remotely from within the **Node**.
- G1347: Secure remote connections to a database.
- G1623: Implement personal **firewall** software on computers used for remote connectivity in accordance with the Desktop Applications, Network, and Enclave **Security Technical Implementation Guides** (**STIGs**).
- G1245: Isolate the **Web service portlet** from platform dependencies using the **Web Services for Remote Portlets** (**WSRP**) Specification protocol.

P1278: Design Tenet: Enterprise Service Management

Considerations

Service Management

- Service management includes tracking the development, deployment, and operation of services. Manage services according to **Node** affiliation using available management services, either **NCES** Enterprise Service Management or local services.
- Expose a service management interface that the node management services can access.

Provisioning of Enterprise Services

- Design the Node's applications and components to enable access to enterprise services as they become available from DoD/DISA.
- When required, implement enterprise services locally at the Node based on technical standards provided by DoD/DISA. When such standards are not specified, choose standards based on best commercial practice.
- Maintain a separable service implementation to enable the replacement of local Node implementations with NCES services as they become available.

Guidance

- G1010: Use **open-standard** logging frameworks.
- G1032: Validate all input fields.
- G1093: Implement exception handlers for **SOAP**-based **Web services**.
- G1094: Catch all exceptions for application code exposed as a **Web service**.
- G1095: Use **W3C** fault codes for all **SOAP** faults.
- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.
- G1155: Use **triggers** to enforce **referential** or data integrity, not to perform complex **business logic**.
- G1209: For Java, use **JDK** logging facilities.
- G1210: For **.NET**, use Debug and Trace from the **`System.Diagnostics` namespace**.
- G1276: Do not modify the contents of the Web browser's status bar.
- G1287: Provide feedback when a transaction will require the user to wait.
- G1639: Describe **Components** exposed by the Node as specified by the **Service Definition Framework**
- G1569: Maintain a comprehensive list of all of the **Components** that are part of the Node.

Best Practices

- BP1868: Incorporate mechanisms to enhance the survivability, resiliency, redundancy, and reliability of Computing Infrastructure (CI).

P1240: Information Assurance/Security

Information assurance (**IA**) refers to measures that protect and defend information and information systems. The goal of IA is to ensure confidentiality, integrity, availability, and accountability by providing capabilities to detect, monitor, react to, and protect against attacks.

Many of the existing solutions to IA problems (and many of the requirements in existing IA regulations) assume that both clients and servers are located on the same physical or logical network. They rely heavily on perimeter or boundary protection. **Service-oriented architecture** (**SOA**) interoperability and loose coupling requirements make those security models inadequate.

In SOA, the boundaries are not clearly defined. Services may be exposed to external clients and not bound to a physical location. The client and service providers may be governed by different security policies.

Base a net-centric IA strategy on a service-level view of security rather than on perimeter security. Developing new security models is necessary to determine how to establish the necessary trust relationships between service requestors and service providers and to select the most adequate and appropriate authentication and authorization mechanisms. To implement a net-centric IA strategy, programs should provide the following:

- Integrated identity management, permissions management, and digital rights management
- Adequate confidentiality, availability, and integrity

Detailed Perspectives

Design Tenet: Net-Centric IA Posture and Continuity of Operations [P1242]
Design Tenet: Identity Management, Authentication, and Privileges [P1243]
Design Tenet: Mediate Security Assertions [P1245]
Design Tenet: Cross-Security-Domains Exchange [P1246]
Design Tenet: Encryption and HAIPE [P1247]
Design Tenet: Employment of Wireless Technologies [P1248]
Other Design Tenets [P1251]

P1242: Design Tenet: Net-Centric IA Posture and Continuity of Operations

This tenet refers to the assignment of Mission Assurance Category (MAC) and Confidentiality Level to a given application, node, or system. The MAC reflects the importance of information relative to the achievement of DoD goals and objectives, particularly the warfighter's combat mission. Mission Assurance Categories primarily determine the requirements for availability and integrity.

There are three defined mission assurance categories:

- MAC I for systems with vital operational needs
- MAC II for systems that are important to deployed or contingency forces
- MAC III for systems supporting day-to-day businesses that do not materially affect support to deployed forces

The complete definitions for those categories are included in DoD Directive 8500.1.[R1197] The security requirement for each combination of mission assurance category and its confidentiality level are in DoD Instruction 8500.2.[R1198]

Considerations

- When assigning a MAC in a net-centric environment, consider not just the intrinsic properties of the node or service, but also its impact on other Information Operations that may call upon it.
- When developing a node or service, account for its potential use by other missions and adjust the MAC appropriately. Incorporate adequate protection and integrity requirements into the design that are commensurate with those potential uses.
- Typically, not all of the potential uses of a node or service are known up front. Therefore, developers must make assumptions about how critical missions may use the node or service when they determine requirements. It may be necessary to modify the MAC to accommodate future, critical missions.

Guidance

- G1634: Certify and accredit **Components** with all applicable DoD **Information Assurance** (IA) processes.
- G1632: Certify and accredit Nodes with all applicable DoD **Information Assurance** (IA) processes.
- G1633: Host only DoD **Information Assurance** (IA) certified and accredited **Components**.
- G1585: Provide a transport infrastructure for the Node that implements **Global Information Grid** (**GIG**) **Information Assurance** (**IA**) boundary protections.

Best Practices

- BP1701: Configure **Components** for **Information Assurance** (IA) in accordance with the Network **Security Technical Implementation Guide** (STIG).
- BP1672: Be prepared to integrate fully with the **Information Assurance** (**IA**) infrastructure.

P1243: Design Tenet: Identity Management, Authentication, and Privileges

Authentication mechanisms are based on **credentials** presented by the requestor. Those credentials may be something the user knows (e.g., passwords), something the user is (e.g., biometrics), something the user has (e.g., smart card), or any combination of these factors.

Each approach is associated with the strength of an authentication. The weakest methods are password-based and the strongest are combinations of biometrics and smart cards.

There are also differing strengths within each method. For instance, systems that require complex passwords are stronger than those that accept simple ones and systems using retina or fingerprint readers are stronger than those that use finger length.

Components that are separate from the implementation of mission- or business-specific functionality often provide **identity management** and authorization.

Identity management is a discipline which encompasses all of the tasks required to create, manage, and delete identities in a computing environment. Some identity management systems available on the market today offer tools to allow one with administrative privileges to assign privileges or authorizations to a particular resource.

Considerations

User Authentication

Authentication normally occurs at the "edge" of an application or node, or at the very first network access. Systems should strive to accept strong authentication methods as early as possible. If possible, migrate authentication tasks to an authentication server and make systems rely on tokens or assertions from the server for authentication. For closed community configurations, these schemes may involve the use of a Kerberos-type single sign-on device.

Identity Management

Use authentication assertions to propagate identities in a secure and trusted way throughout the enterprise. Those assertions should indicate not only the identity and attributes of the requestor, but the strength of the mechanism used to ascertain its identity.

Generate a Trust Model to specify the proper trust relationships and the path for authentication assertions.

Multi-Tier Authentication

While considering the specific method used and its relative strength, remember that in a **Service-Oriented Architecture** (**SOA**) service providers may require stronger authentication than that invoked by the service requestor. These cases may require a multi-tier authentication; i.e., re-authenticating the original requester with the provider by transferring appropriate credentials.

To avoid future multi-tier authentication problems, use strong authentication methods such as PKI certificates whenever possible.

Validation of Authentication Information

A service provider may receive requests that include the original authentication information from the requestor. DoD uses Public Key Infrastructure (PKI) certificates for authentication information. A very effective way for the provider to ascertain the validity of the authentication information is to confirm it through a PKI mechanism.

A service provider, when receiving requestor identification information through a security assertion, must authenticate that an entity that the provider trusts has validated the assertion. PKI signatures provide a means to accomplish this. The signatures must encompass and link both the assertion and the actual request. The service

provider must determine, if using PKI, the complete scheme of how to verify the certificates, the timeliness of the requests, and the current validity of the credential (i.e., verification that the certificates are revoked).

Systems should migrate to PKI authentication as it become available, and start using it as a baseline to provide enterprise authentication services.

Authorization Techniques

Access authorizations are determined by the requester's attributes and by the nature and contents of the request. Make authorization decisions at the access boundary, therefore isolating applications from changes in policy and authorization technology.

Use node-managed security (sometimes referred to as declarative security, programmatic security, or container-managed security), unless application requirements require programmatic authorizations, where individual actions within the service are authorized based on the nature or parameters of the request.

Role-Based Authorizations

Roles are one way to establish authorized access control. In the **Role-Based Access Control** (**RBAC**) environment, role privileges are the basis for access decisions. In RBAC, a trusted entity administers users and their roles in association with the user identity. Roles are typically defined within a system boundary, and occasionally within or between enclaves. Assigning an individual to a role requires that the user be pre-provisioned into the role. Users should never supply a mapping of users to roles directly, but users may select one of multiple roles assigned to them when seeking access to system functionality.

Use the eXtensible Access Control Markup Language (XACML) to retrieve access control information. XACML supports the exchange of access control information using XML. This allows adherence to the principle of least privilege (see the following perspective for additional information on this principle: Apply Principle of Least Privilege [P1317].

Attribute-Based Authorizations

Attribute-Based Access Control (ABAC) is a policy-based, access control solution that uses attributes to enable access. In the ABAC environment, a set of user attributes is the basis for access decisions. These attributes could include, for example, mission function, area of interest, rank, role, citizenship, organization, level of clearance, level of training, and specific assignment location.

When an application retrieves access control information from an external policy decision point (PDP) or retrieves policies for its own resources, it should do so with XACML which supports exchange of access control information using XML. In general, authorization policies should be distinct from application functionality but co-located and co-managed with those applications.

ABAC Advantages

The advantage of ABAC is to enable information sharing to adapt to dynamic changes in the operational environment. For example, one advantage of ABAC is that is can support an authorized but "unanticipated user." Using ABAC concepts, a system administrator can grant access to data through policy based rules using attributes. In this way, information becomes available to unregistered or "unanticipated users." External users with the right attributes have immediate access to relevant information. An external user can discover and gain access to previously "unknown data."
ABAC characteristics in an enterprise can include the following:

- Immediate response to policy change. Applying security policy, through the use of attributes, to resources can reduce the costs and complexities of securely managing individual privileges
- Improved situational awareness. Sharing information on demand when the information is most valuable. ABAC allows for information access rules to be updated due to changes in threat

ABAC and RBAC Relationships

Since ABAC can use a "Role" as an attribute, RBAC can be accomplished using ABAC. It is possible to associate attributes with subjects (such as human users), resources (such as information technology assets), and the environment (such as a threat level, or deployed conditions). User attributes are generally characteristics shared

by large segments of an enterprise's user base, so controlling access via attributes is more flexible and scalable than controlling access by individual user identity.

The scope of the number of systems accessible with a role is only as large as the size of the community within which one can obtain agreement on the definition of roles. Efforts to define standard role definitions across the Services or across Theaters have not resulted in standard, accepted role definitions. Roles can be better defined within **Communities of Interest** (**COI**). Individual COIs can define roles, and the acceptable values to populate roles. For access that must be tightly restricted to those in a particular role, COIs should define and register role definitions and allowable values, and then provision and publish attribute stores that contain role attributes.

ABAC Activities

The DoD and the **Intelligence Community** (**IC**) have joined efforts to develop joint solutions for Authorization and Attribute Services. The DoD and the IC created a joint Authorization and Attribute Services Tiger Team (AATT) in December 2007. The AATT Charter (25 February 2008) provides background information regarding the need to create the AATT. The purpose of the AATT is to identify common interfaces and service specifications that can be used to implement and deploy common authorization and attribute capabilities across the DoD and IC.

These attributes defined by the DoD and IC are stored in the DISA Joint Enterprise Directory Services (JEDS), accessible via Defense Knowledge Online (user registration and PKI certificate required for access).

- Documents and information regarding the AATT are available on DKO at https://www.us.army.mil/suite/page/504666 and on Intellipedia at https://www.intelink.gov/wiki/Authorization_and_Attribute_Tiger_Team
- Information regarding JEDS is available at https://www.us.army.mil/suite/collaboration/folder_V.do?foid=9041194&load=true

Guidance

- G1300: Secure all **endpoints**.
- G1302: Validate all inputs.
- G1306: **Identify** and **authenticate** users of the application.
- G1308: Configure **Public Key Enabled** applications to use a **Federal Information Processing Standard** (**FIPS**) 140-2 certified cryptographic module.
- G1309: Make applications handling high value unclassified information in Minimally Protected environments **Public Key Enabled** to interoperate with **DoD High Assurance** .
- G1310: Protect application cryptographic objects and functions from tampering.
- G1311: Use **Hypertext Transfer Protocol over Secure Socket Layer** (**HTTPS**) when applications communicate with DoD **Public Key Infrastructure** (**PKI**) components.
- G1312: Make applications capable of being configured for use with DoD **PKI**.
- G1313: Provide documentation for application configuration for use with DoD **PKI**.
- G1314: Provide applications the ability to import and export keys (software certificates only).
- G1316: Ensure that applications protect **private keys**.
- G1317: Ensure applications store **Certificates** for subscribers (the owner of the **Public Key** contained in the Certificate) when used in the context of signed and/or encrypted email.
- G1318: Develop applications such that they provide the capability to manage and store **trust points** (**Certificate Authority** Public Key **Certificates**).
- G1319: Ensure applications can recover data encrypted with legacy keys provided by the DoD **PKI** Key Recovery Manager (**KRM**).
- G1320: Use a minimum of 128 bits for **symmetric keys**.
- G1321: Enable applications to be capable of performing **Public Key** operations necessary to verify signatures on DoD **PKI** signed objects.
- G1322: Ensure that applications that interact with the DoD **PKI** using **SSL** (i.e., **HTTPS**) are capable of performing cryptologic operations using the **Triple Data Encryption Algorithm** (**TDEA**).

- G1323: Generate random **symmetric encryption** keys when using symmetric encryption.
- G1324: Protect **symmetric keys** for the life of their use.
- G1325: Encrypt **symmetric keys** when not in use.
- G1326: Ensure applications are capable of producing Secure Hash Algorithm (**SHA**) **digests** of **messages** to support verification of DoD **PKI** signed objects.
- G1327: Enable an application to obtain new **Certificates** for subscribers.
- G1328: Enable an application to retrieve **Certificates** for use, including relying party operations.
- G1330: Ensure applications are capable of checking the status of **Certificates** using a **Certificate Revocation List** (**CRL**) if not able to use the **Online Certificate Status Protocol** (**OCSP**).
- G1331: Ensure applications are able to check the status of a Certificate using the **Online Certificate Status Protocol** (**OCSP**).
- G1333: Only use a **Certificate** during the Certificate's validity range, as bounded by the Certificate's "Validity - Not Before" and "Validity - Not After" date fields.
- G1335: Make applications capable of being configured to operate only with PKI Certificate Authorities specifically approved by the application's owner/managing entity.
- G1338: Ensure that **Public Key** Enabled applications support multiple organizational units.
- G1341: Use a security manager support to restrict application access to privileged resources.
- G1342: Restrict direct access to class internal variables to functions or methods of the class itself.
- G1344: Encrypt sensitive data stored in configuration or resource files.
- G1357: Do not rely solely on transport level security like **SSL** or **TLS**.
- G1362: Validate XML messages against a **schema**.
- G1363: Do not use clear text passwords.
- G1364: Hash all passwords using the combination of a timestamp, a **nonce** and the password for each **message** transmission.
- G1365: Specify an expiration value for all security tokens.
- G1366: Digitally sign all **messages** where non-repudiation is required.
- G1367: Digitally sign **message** fragments that are required not to change during transport.
- G1369: Digitally sign all requests made to a security token service.
- G1371: Use the **Digital Signature Standard** for creating **Digital Signatures**.
- G1372: Use an X.509 **Certificate** to pass a **Public Key**.
- G1373: **Encrypt messages** that cross an **IA** boundary.
- G1374: Individually **encrypt** sensitive **message** fragments intended for different intermediaries.
- G1377: Use **LDAP** 3.0 or later to perform all connections to LDAP repositories.
- G1378: Encrypt communication with **LDAP** repositories.
- G1346: Audit database access.
- G1347: Secure remote connections to a database.
- G1349: Validate all input that will be part of any dynamically generated **SQL**.
- G1350: Implement a strong password policy for **RDBMS**.
- G1351: Enhance database security by using multiple user accounts with constraints.
- G1619: Configure **clients** with a **Common Access Card** (**CAC**) reader.
- G1652: Use DoD **PKI** X.509 **certificates** for **servers**.
- G1380: Use the **XACML** 2.0 standard for **SAML**-based rule engines.
- G1797: Use a minimum of 1024 bits for **asymmetric keys**.

Best Practices

- BP1375: Use **asymmetric encryption** for sensitive **SOAP**-based **Web services**.

P1245: Design Tenet: Mediate Security Assertions

Use security assertions or security tokens to convey user authentication and access authorization to a service provider. Security assertions and tokens are statements that an entity the service provider trusts has generated and validated.

Considerations

Security Assertions

- Use an XML-based standard such as the **Security Assertion Markup Language** (**SAML**) to transfer assertions.
- For close community configurations, start with Kerberos security tokens. Establish implicit trust relationships between entities to circumvent formal validations through the use of trusted channels (e.g., **SSL** transfers).
- Transfer security tokens or security assertions using the general purpose mechanism provided for associating security tokens or assertions with SOAP message contents as specified in the WS-Security Standard. Kerberos and other tokens shall use the Binary Security Token provision. Use SAML assertions in the context of WS-Security as specified in the upcoming WS-Security SAML Token Profile. [R1246]

Chained Requests

- When requests need to be chained (i.e., forwarded to third parties), the security assertions must cover the origin and destination, all intermediate assertions, and the required chain of trust. Earlier request implementations may separate a chained request into separate transactions.

Guidance

- G1379: Use **SAML** version 2.0 for representing security assertions.
- G1380: Use the **XACML** 2.0 standard for **SAML**-based rule engines.
- G1359: Bind **SOAP Web service** security policy assertions to the service by expressing them in the associated **WSDL** file.
- G1357: Do not rely solely on transport level security like **SSL** or **TLS**.
- G1322: Ensure that applications that interact with the DoD **PKI** using **SSL** (i.e., **HTTPS**) are capable of performing cryptologic operations using the **Triple Data Encryption Algorithm** (**TDEA**).

P1246: Design Tenet: Cross-Security-Domains Exchange

Exchange information across security boundaries using air-gap interfaces, electronically enforced one-way interfaces, content-based encryption, content-sensitive security guards, multilevel trusted databases, and multilevel systems. The data exchange may be from low to high or high to low. In an NCW environment, many of the service requests and their corresponding trust assertions may have to cross security boundaries; that is, they must originate and terminate at entities with different security classification levels.

Considerations

Cross-Domain Services

- In a net-centric environment, enterprise-wide services are the most efficient way to handle data exchange transactions and implement cross-domain solutions. Develop special cross-domain services to provide validated resources capable of transferring information between security domains operating at different security classifications. To support net-centric warfare effectively, cross-domain solutions must transition from current models to an agile and flexible, robust and available, trusted yet economical solution set. The most effective method is to provide those services at the enterprise level, compatible with the **Global Information Grid** (**GIG**) and **Net-Centric Enterprise Services** (**NCES**).
- Incorporate the capabilities and procedures of centralized cross-domain solutions as they become available. If possible, systems should demonstrate an evolution towards these enterprise-wide solutions. Rely on existing secure guard solutions or one-way solutions until enterprise-wide solutions are available.

Note: *See the following perspectives for additional considerations: Trusted Guards [P1150] and Cross-Domain Interoperation [P1169].*

Guidance

- G1341: Use a security manager support to restrict application access to privileged resources.
- G1379: Use **SAML** version 2.0 for representing security assertions.
- G1380: Use the **XACML** 2.0 standard for **SAML**-based rule engines.
- G1613: Prepare a **Node** to host new **Component services** developed by other Nodes or by the **enterprise** itself.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.

Best Practices

- BP1698: Plan for the event that **Component** services within a **Node** cannot be invoked across security domains.
- BP1669: Select **XML**-capable **trusted guards**.
- BP1691: Use **Node** implemented **Service Discovery** (**SD**) to meet compartmentalization needs.
- BP1614: Plan a contingency response to the **Node** becoming a new **component service** within another Node.

P1247: Design Tenet: Encryption and HAIPE

Enterprise services must enable secure transmission of identification and role assertions through the use of trusted paths. A trusted path is a communications path where there is confidence alteration of data has not occurred during transport and the data are timely.

> ***Note:*** *The definition of "timely" is not the same for all types of information systems. Services should specify an appropriate definition based on the type of information system (e.g., event-driven, transaction-based) and the type of security threat (e.g., replay attack).*

- Use **Secure Sockets Layer** (**SSL**), Internet Protocol Security (IPSec), or **High Assurance Internet Protocol Encryption** (**HAIPE**) protocols to secure transmission of identification and role assertions in a **TCP/IP** environment. Incorporating message-level encryption may provide additional security.

Guidance

- G1322: Ensure that applications that interact with the DoD **PKI** using **SSL** (i.e., **HTTPS**) are capable of performing cryptologic operations using the **Triple Data Encryption Algorithm** (**TDEA**).
- G1323: Generate random **symmetric encryption** keys when using symmetric encryption.
- G1324: Protect **symmetric keys** for the life of their use.
- G1325: Encrypt **symmetric keys** when not in use.
- G1344: Encrypt sensitive data stored in configuration or resource files.
- G1357: Do not rely solely on transport level security like **SSL** or **TLS**.
- G1363: Do not use clear text passwords.
- G1364: Hash all passwords using the combination of a timestamp, a **nonce** and the password for each **message** transmission.
- G1366: Digitally sign all **messages** where non-repudiation is required.
- G1367: Digitally sign **message** fragments that are required not to change during transport.
- G1369: Digitally sign all requests made to a security token service.
- G1371: Use the **Digital Signature Standard** for creating **Digital Signatures**.
- G1372: Use an X.509 **Certificate** to pass a **Public Key**.
- G1373: **Encrypt messages** that cross an **IA** boundary.
- G1374: Individually **encrypt** sensitive **message** fragments intended for different intermediaries.
- G1376: Do not **encrypt** message fragments that are required for correct **SOAP** processing.
- G1378: Encrypt communication with **LDAP** repositories.
- G1381: Encrypt sensitive persistent data.
- G1320: Use a minimum of 128 bits for **symmetric keys**.
- G1326: Ensure applications are capable of producing Secure Hash Algorithm (**SHA**) **digests** of **messages** to support verification of DoD **PKI** signed objects.
- G1321: Enable applications to be capable of performing **Public Key** operations necessary to verify signatures on DoD **PKI** signed objects.
- G1607: Configure routers according to **National Security Agency** (**NSA**) Router Security Configuration guidance.
- G1797: Use a minimum of 1024 bits for **asymmetric keys**.

Best Practices

- BP1375: Use **asymmetric encryption** for sensitive **SOAP**-based **Web services**.

P1248: Design Tenet: Employment of Wireless Technologies

Considerations

- All data transmissions need integrity assurances that the information has not been altered. For transmission of sensitive or classified information, there should also be assurances that the information has not been exposed to unauthorized users. In the case of wireless technologies, consider those assurances in the context of lack of finite boundaries for information protection, and the possibilities of spoofing (i.e., unauthorized insertions of information). Many standards are being developed for the protection of wireless networks using cryptographic means.
- Systems should encrypt all traffic when using wireless technologies using established standards.

Best Practices

- BP1880: Justify, document, and obtain a waiver for all radio terminal acquisitions that are not JTRS/SCA compliant.

P1251: Other Design Tenets

Provide boundary or perimeter protection for **service-oriented architectures** (**SOAs**) to help prevent penetration from non-DoD external sources. The main defense security regulations, namely DoD 8500 Series and DCID 6/3 [R1247], apply to **SOA** components. Some of the regulations may not directly apply, or they may require special considerations when applied to SOAs.

Considerations

Integrity and Confidentiality

- Encrypt requests and responses to achieve the appropriate level of confidentiality protection using protocols such as the following:
  - **Secure Sockets Layer** (**SSL**) or **Transport Layer Security** (**TLS**) for transport layer security
  - Internet Protocol Security (IPsec) for network layer
  - Secure Multi-purpose Internet Mail Extensions (S/MIME) for email traffic
- Migrate toward message-level encryption using standards such as XML-Encryption and provide message integrity protection using standards such as XML-Digital Signature.
- Include timestamps within messages to prevent recording and playback of messages. All timestamps must use Coordinated Universal Time (UTC), also referred to as Greenwich Mean Time (GMT) or Zulu (Z) time.

Firewall Configurations

- Continue using firewalls and proxy servers to protect the physical boundary of clusters of equipment supporting SOAs. Firewalls must prevent unauthorized penetrations; they require careful programming to reduce the inherent additional risks of SOAs.
- An example of one such risk would be allowing inbound **HTTP**/**HTTPS** access to Web-based applications. This may allow an ill-intended **SOAP** message to cause an internal application buffer overflow while looking completely benign to the firewall. To help prevent such a threat, use XML-capable firewalls as they become available.

Intrusion Detection Systems

- Use adequate monitoring to determine anomalies or failures that can impair mission performance. Intrusion detection systems should detect unauthorized access and penetration attempts. Use detection and protection mechanisms to detect and prevent illicit actions automatically, and complement them with manual reporting of anomalies or specially detected events. Enable automatic reconfiguration or recovery features only for limited and well-defined conditions.

Intrusion Reporting

- A service-oriented architecture requires some centralization of automated reports which, when coupled with correlation and analysis of events detected at multiple nodes, helps establish enterprise security awareness. The scope of the environment conducting the correlation depends on the availability of software agents in individual nodes and the availability of resources that can establish the correlation of events. The scope may range from a few systems at a given location to all activities within a theater of operations. An even broader analysis may occur through manual reporting at an enterprise-wide level.

Audit Events Linkage

- Configure and use individual system audit mechanisms. For SOAs, complement audits with mechanisms that correlate events in different nodes and provide network-wide forensics. Time stamping and logging of all inter-node messages help link events and actions involving multiple nodes. Use UTC for time stamping.

Use of Audits for Attribution

- Use logging and request auditing to satisfy attribution requirements (i.e., determination of the individual responsible for the action). This should occur at both the requestor and service provider sites.

GIG Policy Compliance

- Develop systems in accordance with the IA requirements in DoD Instruction 8500.2 [R1198] for the appropriate Mission Assurance Category and Sensitivity Level. Systems dealing with intelligence sources and methods must also comply with DCID 6/3. [R1247] Also leverage the guidance and technologies described in DoD CIO Guidance and Policy Memorandum 6-8510, ***DoD GIG Information Assurance*** [R1251] and the ***End-to-End Information Assurance of the GIG***. [R1252]

Certification and Accreditation

- Certify and accredit all systems in accordance with DoD Instruction 8510.01, ***DoD Information Assurance Certification and Accreditation Process (DIACAP)***. [R1291] In addition, Air Force systems should comply with the certification and accreditation section in Air Force Instruction 33-202, ***Network and Computing Security***. [R1249]

Guidance

- G1301: Practice layered security.
- G1302: Validate all inputs.
- G1359: Bind **SOAP Web service** security policy assertions to the service by expressing them in the associated **WSDL** file.
- G1363: Do not use clear text passwords.
- G1364: Hash all passwords using the combination of a timestamp, a **nonce** and the password for each **message** transmission.
- G1365: Specify an expiration value for all security tokens.
- G1369: Digitally sign all requests made to a security token service.
- G1372: Use an X.509 **Certificate** to pass a **Public Key**.
- G1376: Do not **encrypt** message fragments that are required for correct **SOAP** processing.
- G1339: Practice defensive programming by checking all method arguments.
- G1340: Log all exceptional conditions.
- G1346: Audit database access.
- G1348: Log database **transactions**.
- G1349: Validate all input that will be part of any dynamically generated **SQL**.
- G1622: Implement **commercial off-the-shelf** (**COTS**) software that protects against malicious code on each operating system in the Node in accordance with the Desktop Application **Security Technical Implementation Guide** (**STIG**).
- G1623: Implement personal **firewall** software on computers used for remote connectivity in accordance with the Desktop Applications, Network, and Enclave **Security Technical Implementation Guides** (**STIGs**).
- G1624: Install anti-**spyware** software on all Windows Desktop computers.
- G1632: Certify and accredit Nodes with all applicable DoD **Information Assurance** (IA) processes.
- G1633: Host only DoD **Information Assurance** (IA) certified and accredited **Components**.
- G1634: Certify and accredit **Components** with all applicable DoD **Information Assurance** (IA) processes.
- G1662: Follow the guidance provided in the **Security Technical Implementation Guide** (**STIG**) for **Domain Name System** (**DNS**) implementations.
- G1667: Implement **Virtual Private Networks** (**VPNs**) in accordance with the guidance provided in the Network **Security Technical Implementation Guide** (**STIG**).

P1241: Transport

The Transport Infrastructure is a foundation for net-centric transformation in DoD. To realize the vision of the **Global Information Grid** (**GIG**), the Assistant Secretary of Defense for Networks and Information Integration/DoD Chief Information Officer (ASD(NII)/DoD CIO) has called for a dependable, reliable, and ubiquitous network that eliminates stovepipes and responds to the dynamics of the operational scenario. To construct the Transport Infrastructure, DoD will do the following:

- Follow the **Internet** model
- Create the GIG from smaller component building blocks
- Design with interoperability, flexibility to evolve, and simplicity in mind
- Provide a common, black-core IP network for both unclassified and encrypted classified information

Both users and providers of transport services must conform to established and evolving transport-related standards and guidelines. The DoD IT Standards Registry (DISR) [R1179] is the primary source for DoD-adopted standards.

> ***Note:*** *See the Node Transport [P1138] perspective for further guidance.*

- Design Tenet: IPv6 [P1255]
- Design Tenet: Packet Switched Infrastructure [P1260]
- Design Tenet: Layering and Modularity [P1261]
- Design Tenet: Transport Goal [P1262]
- Design Tenet: Network Connectivity [P1263]
- Design Tenet: Concurrent Transport of Information Flows [P1264]
- Design Tenet: Differentiated Management of Quality-of-Service [P1265]
- Design Tenet: Inter-Network Connectivity [P1266]
- Design Tenet: DoD IT Standards Registry (DISR) [P1267]
- Design Tenet: RF Acquisition [P1269]
- Design Tenet: Joint Net-Centric Capabilities [P1274]
- Design Tenet: Operations and Management of Transport and Services [P1277]

P1255: Design Tenet: IPv6

In the next few years, the adoption of **IPv6** throughout the DoD and other Federal Agencies will pass a major implementation threshold. Most DoD bases and other facilities will be IPv6 capable. Most of the key components of the technology are in place for native deployment of IPv6 or dual existence of **IPv4** and IPv6.

A 9 June 2003 ASD(NII)/DoD CIO memo, *Internet Protocol Version 6 (IPv6)*, is the first in a series of memos addressing DoD transition to IPv6 [R1190]. The main points of the directives follow:

- The original goal for IPv6 transition completion was FY08.
- DoD is conducting enterprise-wide deployment of IPv6 in a controlled, integrated and cohesive manner (see the DoD IPv6 Transition Plan [R1205]).
- The DoD IPv6 Transition Office established within **DISA** is responsible for coordinating transition efforts, providing required infrastructure, and insuring that unified solutions are used across DoD. Each Service has a Transition Office responsible for providing technical guidance and transition governance to programs. This includes developing transition plans (subject to coordination into a master plan by DISA), dispensing IP addresses originating from DISA, implementing waiver policy, etc.
- A mandate, to minimize costs of transition, is that all **GIG** assets being developed, procured or acquired must be IPv6 capable (in addition to maintaining interoperability with IPv4 capabilities). The DoD CIO directives contain an outline for the "IPv6 capable" requirement, while a detailed specification is still under development.
- The transition to IPv6 should be accomplished through the normal technical refresh cycle whenever possible.

Considerations

Support IPv6 Transition

- Be able to interoperate with interfacing transport service providers who use either IPv6 or IPv4 during the transition from IPv4. New applications should be IP version agnostic and shall employ an operating system that supports both IPv4 and IPv6. For existing IPv4 service users, the governing authority (e.g., Component IPv6 Transition Office) should develop and approve IPv6 migration plans.
- Transport service providers interfacing with non-transitioned networks must support both IPv6 and IPv4 during the transition from IPv4. Mechanisms proposed to allow the two protocols to coexist and inter-operate during the transition phase from IPv4 to IPv6 include the following:
  - Incorporating both IPv4 and IPv6 support in routers and computers; this is called ***dual stacking***. This is a preferred way to ensure the interoperability between systems during the transition period.
  - Transporting IPv6 traffic through IPv4 networks by encapsulating IPv6 packet in IPv4 and vice-versa; this is called ***tunneling***. During the initial enabling of IPv6 in operational environments in controlled enclaves, tunneling becomes a useful communication mechanism between the enclaves. Tunneling should be considered only as a temporary solution.
  - Placing translation gateways between IPv4 and IPv6 networks or hosts. This is the only mechanism allowing a native IPv4-only device to communicate with IPv6-only device. The expectation is that these devices will not be needed until the later stages of transition for dominant IPv6 devices to communicate with some lingering native IPv4 legacy devices. [R1255]
- In all cases, coordinate IPv6 transport provider planning with the Service IPv6 Transition Office.

Support IPv6 IP security features for data integrity and confidentiality.

- IPv6 provides improved security features in comparison to IPv4 through IPSec and mandatory support for end-to-end security. The Service Transition Office should be able to provide guidance on utilizing any of the IPv6 security features in the context of the service enterprise transition plan.
- Implement DoD-adopted IPv6 standards and products. The list of standards directly relevant to DoD and approved for the use on DoD networks is maintained in the DISR. [R1179]

Guidance

- G1586: Provide a transport infrastructure for the Node that is **Internet Protocol Version 6** (**IPv6**) capable in accordance with the appropriate governing transition plan.
- G1587: Prepare an **Internet Protocol Version 6** (**IPv6**) transition plan for the Node.
- G1588: Coordinate an **Internet Protocol Version 6** (**IPv6**) transition plan for a Node with the **Components** that comprise the Node.
- G1589: Address issues in the appropriate governing **Internet Protocol Version 6** (**IPv6**) transition plan as part of the IPv6 Transition Plan for a Node.
- G1590: Include transition of all the impacted elements of the network as part of the **Internet Protocol Version 6** (**IPv6**) Transition Plan for a Node.
- G1591: Prepare IPv6 Working Group products as part of the **Internet Protocol Version 6** (**IPv6**) transition plan for a Node.
- G1592: Include interoperability testing in the plan as part of the **Internet Protocol Version 6** (**IPv6**) transition plan for a Node.
- G1599: Simultaneously support **Internet Protocol Version 4** (**IPv4**) and **Internet Protocol Version 6** (**IPv6**) in the Node's **Domain Name System** (**DNS**) service.
- G1600: Obtain **Internet Protocol Version 6** (**IPv6**) addresses to use for DoD IP addressable resources from **DISA**.
- G1595: Implement **Domain Name System** (**DNS**) to manage hostname/address resolution within the Node.

Best Practices

- BP1863: Make shareable data assets visible, even if they are not accessible.
- BP1705: Design **Domain Name System** (**DNS**) infrastructure in accordance with appropriate governing **Internet Protocol Version 6** (**IPv6**) Transition Office requirements.
- BP1663: Design a **Domain Name System** (**DNS**) in coordination with the appropriate governing **Internet Protocol Version 6** (**IPv6**) Transformation Office.

P1260: Design Tenet: Packet Switched Infrastructure

The **Global Information Grid** (**GIG**) includes a number of component networks. Each must pass data both internally among its network members and externally to or from other GIG components. As such, the design of the Internet model that applies to the development of the GIG transport infrastructure needs to be an IP datagram delivery system consisting of a packet-switched communications facility in which a number of distinguishable component networks (including any networks external to this system) are connected together using routers. Technologies such as routing standards and quality of service (QoS) mechanisms are needed to achieve the end-to-end functionality the GIG requires. Design and apply these within the framework of packet-switched transport infrastructure.

Considerations

- Implement interface(s) to one and only one network layer protocol (Layer-3 in the *OSI Reference Model*) for datagrams. This applies to transport service providers and consumers and to datagrams passed within a component network and those destined for external networks. The fundamental goal is a single inter-network protocol.
- GIG component system designers should consider how the component transport infrastructure will accept externally-generated IP datagrams that are destined for hosts inside their system. This allows their system to "attach" to the GIG. The designers should also consider how their component infrastructure will deliver internally generated IP datagrams to hosts outside their system, and how it will serve as a transit network for externally generated IP datagrams.

Guidance

- G1595: Implement **Domain Name System** (**DNS**) to manage hostname/address resolution within the Node.
- G1596: Use **Domain Name System** (**DNS**) **Mail eXchange (MX) Record** capabilities to configure electronic mail delivery to the Node.
- G1598: Allow dynamic **Domain Name System** (**DNS**) updates to the Node's internal DNS service by local **Dynamic Host Configuration Protocol** (**DHCP**) **server(s)**.
- G1601: Use configurable **routers** to provide dynamic **Internet Protocol** (**IP**) address management using the **Dynamic Host Configuration Protocol** (**DHCP**).
- G1602: Use configurable **routers** to provide static **Internet Protocol** (**IP**) addresses.
- G1604: Use configurable **routers** to provide time synchronization services using **Network Time Protocol** (**NTP**).
- G1605: Use configurable **routers** to provide **multicast** addressing.
- G1606: Manage **routers** remotely from within the **Node**.
- G1607: Configure routers according to **National Security Agency** (**NSA**) Router Security Configuration guidance.
- G1608: Obtain reference time from a standard globally synchronized time source.
- G1609: Arrange for a backup time source.
- G1610: Configure the **Dynamic Host Configuration Protocol** (**DHCP**) services to assign **multicast** addresses.
- G1611: Implement Internet Protocol (**IP**) gateways to interoperate with the **Global Information Grid** (**GIG**) until IP is supported natively for **Components** that are not IP networked.

Best Practices

- BP1864: Layer architectures to support clear boundaries between data management, presentation, and business logic functionality.
- BP1877: Align end-to-end interoperable management of **QoS** with external networks.
- BP1878: Quantitative measures of QoS requirements should be supportable.
- BP1879: The program, project or initiative should align with the DoD QoS/CoS Working Group Roadmap.
- BP1876: Provide a priority-based differentiated management of **quality-of-service** for traffic based on class of user, application, or mission.

P1261: Design Tenet: Layering and Modularity

Change is probably the only inviolable characteristic of the commercial **Internet** model. Moreover, change occurs at different rates in different elements of the network/protocol stack. Design the **Global Information Grid** (**GIG**) transport infrastructure to accommodate that change. The most effective way to allow differential change in a system is through modular, layered design.

Although market forces and commercial practice sometimes have deprecated the International Organization for Standardization (ISO) Open System Interconnection (OSI) Model, it still provides excellent guidelines for implementing a layered design. These guidelines still apply to the development of the GIG transport infrastructure.

In a layered design, each layer is independent and adds value to the set of services offered by lower layers. The services provided to and from a layer are well defined; however, the precise approach for providing these services is not specified. ISO defined a number of principles to consider when developing a layered design and applied those principles to develop the seven-layer OSI Model.

While a seven-layer approach may not be the solution for the GIG transport infrastructure, GIG component system designers should consider the principles ISO defined to facilitate interoperability and to reduce technology interdependencies that add to system complexity. The following considerations include a subset of these principles that apply to the GIG transport infrastructure.

Considerations

Define Layer Boundaries and Interfaces

- Implement one or more interfaces to the defined transport service delivery point(s) or interface boundaries, where the services description can minimize the number of interactions across the interface boundary(ies). The networks should provide the interface boundary definition(s). To the maximum extent possible, functionality implemented within each OSI layer of the transport service implementation should only interface with the adjacent lower layer via defined interfaces. The goal is to minimize the cross-layer physical and functional interdependencies to facilitate GIG transport infrastructure growth and interoperability.

Ensure Functions are Modular and Separable

- Create a layer of easily localized functions. These functions should enable developers to totally redesign the layer and its protocols to take advantage of new advances in architectural, hardware, or software technology without changing the services and interfaces with the adjacent layers.
- Identify all instances in the transport infrastructure where a logical or physical coupling or dependency exists between different layers of the protocol stack. The goal is to minimize the cross-layer physical and functional interdependencies to facilitate GIG transport infrastructure growth and interoperability.

Minimize Complexity of Layered Implementation

- Keep the number of layers within networks small enough to reduce the complexity of describing, integrating, and maintaining the layers.

Guidance

- G1301: Practice layered security.

Best Practices

- BP1876: Provide a priority-based differentiated management of **quality-of-service** for traffic based on class of user, application, or mission.
- BP1790: Stipulate that the Offeror is to describe how the proposed technical solution reuses services or demonstrates composeability and extensibility by building from existing reusable components and/or services.
- BP1829: Use the **Data Distribution Service** (DDS) **`OWNERSHIP`** **Quality of Service** (QoS) kind set to **`EXCLUSIVE`** when multiple **DataWriters** cannot write each unique data-object within a DDS **Topic** simultaneously.

P1262: Design Tenet: Transport Goal

A design goal of the **Global Information Grid** (**GIG**) is network convergence with voice, video, and other multimedia traffic packetized and transported along with data traffic over a common **Internet Protocol** (**IP**) network. Another transport goal is the convergence of encrypted classified information flows on a common black IP network. This corresponds to the direction of commercial industry, where telecommunications providers and corporate telephony are migrating to IP.
A primary benefit of convergence is that it eliminates the expensive hardware and complexity of separate, dedicated networks that support serial-based traffic (e.g., voice and video teleconferencing). Other benefits include greater efficiency of bandwidth and the ability to introduce new features based on converged services.

Considerations

Support Interfaces with Converged Traffic Networks

- Implement interfaces to, or transition to, a transport infrastructure supporting full convergence of traffic on a single IP inter-network, using DoD-adopted standards and DISA/**JITC**-certified (voice) solution sets.
- Identify and minimize all instances where performance standards cannot be met using a converged transport infrastructure (e.g., where dedicated, single-traffic-type transport service is required). The goal is to minimize cross-layer physical and functional interdependencies to facilitate GIG transport infrastructure growth and interoperability.
- Voice, video, and other multimedia traffic have relatively strict delivery requirements with regard to latency and jitter. This requires networks to support the QoS features identified in the Design Tenet: Differentiated Management of Quality-of-Service [P1265].
- The DoD-adopted set of standards appears in the DoD IT Standards Registry (DISR) [R1179]. DISR specifies standards for Voice over IP (VoIP) and video teleconferencing (VTC) based on the **International Telecommunication Union** (**ITU**) standard H.323.
- Voice over IP (VoIP) refers to a set of standards and technologies that allow transmission of voice data over IP networks. The industry has embraced two different sets of standards:
  - ITU H.323 is the more mature and complete set of standards, which encapsulates Integrated Services Digital Network (ISDN) call signaling over an IP-based network.
  - A more recent set of standards, developed by the **Internet Engineering Task Force** (**IETF**), is based on the Session Initiation Protocol (SIP). The SIP standard concerns simple call placement and is designed to be easily expandable.
- Since there are currently two options for VoIP, the DoD plans to select a set of mandated standards within the DISR.
- Video teleconferencing over IP is based on ITU H.323. This is an umbrella standard of ITU recommendations that address audio, video, signaling, and control for packet-switched networks.

Guidance

- G1585: Provide a transport infrastructure for the Node that implements **Global Information Grid** (**GIG**) **Information Assurance** (**IA**) boundary protections.
- G1584: Provide a transport infrastructure that is shared among **components** within the Node.
- G1586: Provide a transport infrastructure for the Node that is **Internet Protocol Version 6** (**IPv6**) capable in accordance with the appropriate governing transition plan.

Best Practices

- BP1864: Layer architectures to support clear boundaries between data management, presentation, and business logic functionality.
- BP1875: Describe the process and protocols used to provide concurrent traffic from multiple security domains on a single **IP** internetwork.

- BP1877: Align end-to-end interoperable management of **QoS** with external networks.
- BP1878: Quantitative measures of QoS requirements should be supportable.
- BP1879: The program, project or initiative should align with the DoD QoS/CoS Working Group Roadmap.
- BP1594: Examine the use of **Transmission Control Protocol** (**TCP**) extensions and other transport protocols that have been designed to mitigate risk for high bandwidth, high latency satellite communications.
- BP1876: Provide a priority-based differentiated management of **quality-of-service** for traffic based on class of user, application, or mission.

P1263: Design Tenet: Network Connectivity

Provide network connectivity to all end points, such as wide- and local-area networks, and direct connections to mobile end users. This perspective addresses the Open System Interconnection (OSI) Model Layer-2 or terminal-to-network interfaces.

Considerations

Manage Scalability and Complexity

- Quantitatively evaluate scalability before formulating a final design. The evaluation should identify any transport infrastructure design drivers regarding the number of hosts that need to be supported and/or number of networks that are required to support the technologies chosen for the specific transport service or infrastructure use.
- One way to reduce complexity is to use a minimal set of standards/protocols in developing the **Global Information Grid** (**GIG**) transport infrastructure. This implies that any selected standard/protocol has the capacity to serve as large a percentage of the GIG as possible. Component systems of the GIG should select standards/protocols that can scale to the enterprise. GIG component system designers should evaluate their transport infrastructure design to identify any instances where different technology/protocols perform the same function (e.g., internal routing).

Optimize Use of COTS Products

- Use open, **commercial-off-the-shelf** (**COTS**) products as much as possible. Government-off-the-shelf (GOTS) and/or vendor-unique products may lead to interoperability and evolvability issues. Use them only when there is an overarching, unique, DoD requirement driving that selection.
- Document the justification for the use of any protocols, standards, etc., that are not included in the DoD IT Standards Registry and/or could not be purchased off-the-shelf from a commercial networking vendor.

Guidance

- G1605: Use configurable **routers** to provide **multicast** addressing.
- G1606: Manage **routers** remotely from within the **Node**.
- G1602: Use configurable **routers** to provide static **Internet Protocol** (**IP**) addresses.
- G1601: Use configurable **routers** to provide dynamic **Internet Protocol** (**IP**) address management using the **Dynamic Host Configuration Protocol** (**DHCP**).
- G1604: Use configurable **routers** to provide time synchronization services using **Network Time Protocol** (**NTP**).
- G1609: Arrange for a backup time source.
- G1607: Configure routers according to **National Security Agency** (**NSA**) Router Security Configuration guidance.
- G1330: Ensure applications are capable of checking the status of **Certificates** using a **Certificate Revocation List** (**CRL**) if not able to use the **Online Certificate Status Protocol** (**OCSP**).
- G1610: Configure the **Dynamic Host Configuration Protocol** (**DHCP**) services to assign **multicast** addresses.
- G1608: Obtain reference time from a standard globally synchronized time source.
- G1582: In Node **Enterprise Service** schedules, include version numbers of standard Enterprise Services interfaces being implemented.

Best Practices

- BP1830: Use the **Data Distribution Service** (DDS) Content Profile to tailor subscription message data.
- BP1651: Ensure **Node Components** have access to **Core Enterprise Services**.
- BP1845: Consider key enterprise-level concerns when planning and executing a migration to net-centricity and **SOA**.

P1264: Design Tenet: Concurrent Transport of Information Flows

This tenet addresses the use of Inline Network Encryptors (INEs) that allow all security domains to be "known" globally to the Open System Interconnection (OSI) Model Layer-3 encrypted backbone network. This is a fundamental shift from current link-by-link encryption. Utilizing a Black Core [P1152] network should provide a significantly streamlined communications infrastructure that also makes more efficient use of the available bandwidth through the invocation of quality-of-service/class-of-service (QoS/CoS) based IP datagram multiplexing.

High Assurance Internet Protocol Encryptor (**HAIPE**) devices are among the critical technologies that should enable the Black Core IP-network vision to become a reality. However, a number of technical challenges must be solved before the vision can be realized across all functional domains and **Communities of Interest** (**COIs**). These include the following:

- Support for IP-based QoS/CoS
- Support for dynamic unicast IP routing
- Support for dynamic multicast IP routing
- Support for mobility
- Support for simultaneous **IPv6** and **IPv4** operation

Considerations

Implement INE Standards and Products to Support Traffic Convergence

- Government-off-the-Shelf (GOTS) and/or vendor-unique products may lead to interoperability and evolvability issues. Use them only when there is an overarching, unique, DoD requirement driving that selection.
- Implement DoD-adopted INE standards and products, when available, to support traffic convergence from multiple security domains on a single IP inter-network. Currently, DoD is engaged in **Internet Engineering Task Force** (**IETF**) standards working groups and vendor communities to accelerate development of new standards in the areas of security, tactical communications, QoS, and reliable networking. Some standards have been adopted for QoS and HAIPE. A product list is in development for infrastructure, hardware, software, and other categories of IPv6 products.

Document Approach to Information Infrastructure with Black Core

- GOTS and/or vendor-unique products may lead to interoperability and evolvability issues. Use them only when there is an overarching, unique, DoD requirement driving that selection.
- Document the approach to providing an information infrastructure with a Black Core.

Guidance

- G1607: Configure routers according to **National Security Agency** (**NSA**) Router Security Configuration guidance.

Best Practices

- BP1875: Describe the process and protocols used to provide concurrent traffic from multiple security domains on a single **IP** internetwork.
- BP1879: The program, project or initiative should align with the DoD QoS/CoS Working Group Roadmap.
- BP1880: Justify, document, and obtain a waiver for all radio terminal acquisitions that are not JTRS/SCA compliant.
- BP1670: Plan for Black Core implementation in the local Node.
- BP1671: Consider Black Core transition whenever there is a significant Node network design or configuration decision to make in an effort to avoid costly downstream changes caused by Black Core transition.

P1265: Design Tenet: Differentiated Management of Quality-of-Service

Some applications in the **Global Information Grid** (**GIG**) require firm service guarantees, while others operate correctly if they receive services that are differentiated with respect to one or more performance characteristics.

Differentiated Services or DiffServ aggregates flows into coarse classes and then treats the packets in these classes differentially. Due to this aggregation, and the resulting absence of a need to consider individual flows beyond the edges of an internet, DiffServ exhibits good scaling properties. However, in the absence of additional mechanisms, DiffServ provides only preferential, differentiated levels of service and not guarantees.

Considerations

Support Quality of Service (QoS) and Class of Service (CoS)

- Interoperate with interfacing transport service providers who use standardized DoD QoS/CoS in accordance with the DoD QoS/CoS Roadmap. As the interfacing networks are transitioned to standardized QoS/CoS, plan to migrate to maintain interoperability.
- Prioritize traffic based on class of user, application, or mission. Lower priority data flows should be preempted if a higher priority flow is initiated and insufficient resources exist to carry both flows simultaneously. This capability, referred to as Class of Service (CoS) support, corresponds approximately to the notion of Multi-Level Priority and Preemption (MLPP). The GIG and its components should support both QoS and CoS in accordance with the DoD QoS/CoS Roadmap and policies

Guidance

- G1771: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of a **publisher**.
- G1801: Explicitly define a **Topic Quality of Service** (QoS) for each **Data Distribution Service** (DDS) Topic within a DDS **Domain**.
- G1803: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe real-time messaging criteria for **Publishers**.
- G1804: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe **DataWriter**.
- G1805: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of the **Subscriber**.
- G1806: Explicitly define the Request-Offered **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of the **DataReader**.
- G1808: Handle all **Data Distribution Service** (DDS) **Quality of Service** (QoS) contract violations using one of the **Subscriber access APIs**.

Best Practices

- BP1876: Provide a priority-based differentiated management of **quality-of-service** for traffic based on class of user, application, or mission.
- BP1877: Align end-to-end interoperable management of **QoS** with external networks.
- BP1878: Quantitative measures of QoS requirements should be supportable.
- BP1879: The program, project or initiative should align with the DoD QoS/CoS Working Group Roadmap.

P1266: Design Tenet: Inter-Network Connectivity

A fundamental tenet of the commercial Internet model is that the complexity of the Internet belongs at the edges. Certain required end-to-end functions can only be performed correctly by the end systems themselves. Any network, however carefully designed, will be subject to failures of transmission at some statistically determined rate.

The best way to cope with this is to accept it and give responsibility for the integrity of communication to the end systems. This principle drives the complexity of the network to the edge and limits state information held inside the network. This increases the robustness of end-to-end communications since application state can now only be destroyed by a failure of the end systems.

Many issues need to be resolved to mature the guidance for this tenet, especially for transport users whose data traverse different media with different performance characteristics. In some situations it may not be desirable to follow this design tenet.

For example, the use of **Transmission Control Protocol** (**TCP**) proxies, which may be required to achieve adequate performance across satellite assets, runs counter to this tenet. The proxy (part of the network and not an end system) maintains state information on the TCP **session** between two end-user systems, but it cannot guarantee that the function that TCP is performing is being accomplished.

Avoid implementing "intelligence" within the network whenever possible.

Considerations

Support Inter-network Connectivity Using DoD-Adopted Standards

- Support inter-network connectivity using DoD-adopted standard protocols contained in the ***DoD IT Standards Registry (DISR)*** [R1179], such as BGP4. Any protocols or standards that are not included in the DISR, such as performance-enhancing proxies, should be documented and justified against the resulting impact to GIG component system interoperability.

Guidance

- G1601: Use configurable **routers** to provide dynamic **Internet Protocol** (**IP**) address management using the **Dynamic Host Configuration Protocol** (**DHCP**).
- G1602: Use configurable **routers** to provide static **Internet Protocol** (**IP**) addresses.
- G1604: Use configurable **routers** to provide time synchronization services using **Network Time Protocol** (**NTP**).
- G1605: Use configurable **routers** to provide **multicast** addressing.
- G1606: Manage **routers** remotely from within the **Node**.
- G1607: Configure routers according to **National Security Agency** (**NSA**) Router Security Configuration guidance.
- G1608: Obtain reference time from a standard globally synchronized time source.
- G1609: Arrange for a backup time source.
- G1610: Configure the **Dynamic Host Configuration Protocol** (**DHCP**) services to assign **multicast** addresses.
- G1623: Implement personal **firewall** software on computers used for remote connectivity in accordance with the Desktop Applications, Network, and Enclave **Security Technical Implementation Guides** (**STIGs**).

P1267: Design Tenet: Joint Technical Architecture [now DISR]

> ***Note:*** *This topic is "Design Tenet: Joint Technical Architecture" in the Net-Centric Checklist v2.1.3 of 12 May 2004. The DISR Baseline Release 04-2.0 of 22 December 2004 replaced the JTA so this perspective refers to the DISR rather than the JTA.*

DoD-approved standards and protocols related to net-centricity are in the *DoD Information Technology (IT) Standards Registry* (DISR).[R1179] Programs, projects or initiatives should support computing infrastructure that is compliant with the net-centric interoperability standards in the DISR. NESI provides implementation guidance and best practices for DoD sanctioned standards and protocols. However, other standards are often useful and when a program (or project or initiative) uses them, the program manager needs to be able to justify this use. Many of the technologies and implementation specifics associated with the **ASD(NII)** *Net-Centric Checklist* Tenets are still in development and have not yet reached maturity.

Considerations

- Justify and document all standards that are not included in the DISR,[R1179] especially those that impact transport service infrastructure design.

Best Practices

- BP1712: Register developed mappings in the **DoD Metadata Registry**.
- BP1875: Describe the process and protocols used to provide concurrent traffic from multiple security domains on a single **IP** internetwork.

P1269: Design Tenet: RF Acquisition

Considerations

JTRS/SCA Compliance

- Justify, document, and obtain a waiver for all radio terminal acquisitions that are not **Joint Tactical Radio System** (**JTRS**) /**Software Communications Architecture** (**SCA**) compliant and coordinate with the Office of the Secretary of Defense (OSD) and the JTRS Joint Program Executive Office (JPEO); the following references apply: [R1240] and [R1241].

Minimize RF Bandwidth Requirements

- Use appropriate transmit protocols, compression standards, and other techniques when interfacing radio frequency (RF) networks to the **Global Information Grid** (**GIG**) environment. The RF environment, with its much more constrained and error prone propagation environment, requires techniques that minimize bandwidth requirements.

Guidance

- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.

Best Practices

- BP1715: Design SCA log services according to the OMG Lightweight Log Service Specification.

P1274: Design Tenet: Joint Net-Centric Capabilities

The Assistant Secretary of Defense for Networks and Information Integration/Department of Defense Chief Information Officer (ASD[NII]/DoD CIO) issued a 15 July 2003 memorandum, *Joint Net-Centric Capabilities*,[R1258] that identifies a number of key **C4ISR** programs for integrating into the **Global Information Grid** (**GIG**):

- All Space Terminal acquisitions
- All Intelligence, Surveillance, and Reconnaissance (ISR) programs
- Teleport
- Warfighter Information Network-Tactical (WIN-T)
- All radio and data link applications
- Global Command and Control System (GCCS, Joint and Service variants)
- Crypto Modernization
- Distributed Common Ground Systems (DCGS)
- All C2 programs
- Deployable Joint Command and Control (DJC2)
- **High Assurance Internet Protocol Encryption** (**HAIPE**)
- Future Combat Systems (FCS)
- Programs under the FORCEnet umbrella

The memo highlights programs that are required to develop transition plans for integrating transport components with the following GIG joint net-centric capabilities:

- **Internet Protocol Version 6** (**IPv6**)
- **Net-Centric Enterprise Services** (**NCES**)
- **Joint Tactical Radio System** (**JTRS**)/**Software Communications Architecture** (**SCA**)
- Global Information Grid Bandwidth Expansion (GIG-BE)
- Transformational Communications Satellite/Advanced Wideband System
- End-to-end information assurance

The **ASD(NII)** *Net-Centric Checklist* [R1177] also highlights the need for the programs to include in transition plans the use of guard technologies, and standards and protocols for connectivity with allied and coalition partners.

- Use the Joint Chiefs of Staff Instruction (CJCSI) 6212.01E, *Interoperability and Supportability of Information Technology and National Security Systems*, 15 December 2008, to guide implementation of Joint net-centric capabilities.

> ***Note:*** *CJCSI 6212.01E removed the (**NCOW RM**) element of the **Net-Ready Key Performance Parameter** (**NR-KPP**), integrating the components of the former NCOW RM into other elements of the NR-KPP.*

Guidance

- G1629: Identify which **Net-Centric Enterprise Services** (**NCES**) capabilities the Node requires during deployment.
- G1576: Provide an environment to support the development, build, integration, and test of net-centric capabilities.

Best Practices

- BP1866: Coordinate with end users to develop interoperable materiel in support of high-value mission capability.
- BP1880: Justify, document, and obtain a waiver for all radio terminal acquisitions that are not JTRS/SCA compliant.
- BP1681: Make metrics for **component** services visible and accessible as part of the service registration and update the metrics periodically.

- BP1840: Identify opportunities to apply the principles of net-centricity and **SOA** throughout the course of the program.
- BP1661: Engage with the **Net-Centric Enterprise Services** (**NCES**) program office to explore approaches for mobile use of the **Core Enterprise Services** (**CES**) services in mobile Nodes that rely on **Transmission Control Protocol/Internet Protocol** (**TCP/IP**) for inter-node communication.
- BP1837: Update the **net-centric** and SOA migration plan in an iterative manner as the program gains migration experience and conditions change.
- BP1400: Programs will use authoritative **metadata** established by the Joint Mission Threads (JMTs) when available.
- BP1686: Align Node interfaces to **Components** for directory services with the guidance being provided by the Joint Directory Services Working Group (JDSWG) and sub-working groups, including such guidance as naming conventions, federation, and synchronization.

P1277: Design Tenet: Operations and Management of Transport and Services

This tenet encompasses three equally important principles of **Network Operations** (**NetOps**):

- Develop manageable systems
- Use non-proprietary implementations
- Use accepted industry standards

NetOps:

- Is a coordinated, comprehensive set of operational concepts and structure that fuses Systems and Network Management, Information Assurance/Computer Network Defense, and Content Staging/Information Dissemination Management into a single integrated operational construct
- Is an end-to-end capability that represents the integrated doctrine, force structure, and tactics, techniques, and procedures (TTP) needed to manage and direct the net-centric operations of the **Global Information Grid** (**GIG**)
- Encompasses all activities directly associated with the net-centric management and protection of GIG computing (including applications and systems), communications, and information assurance assets across the continuum of military operations
- Actively integrates those capabilities with the goal of end-to-end, assured network availability, information delivery, and information protection

Considerations

Develop Manageable Systems

- Build transport communications and network systems, services, subsystems, sub-services, components, devices, and elements from the ground up to be "manageable." They should also have the appropriate functional management capabilities.
- Manage transport communications and network services and systems proactively and operate to specific levels of service. These service levels are documented and published in Operational or Service Level Agreements (OLA/SLAs).
- Fully integrate management solutions for transport systems and services with management solutions to ensure that the GIG is holistically operated and managed to support operational warfighter requirements. Operational management solutions should fully address all specific management functional areas; e.g., fault, configuration, accounting, performance, and security management.

Use Non-Proprietary Implementations

- Base operational management capabilities and solutions on non-proprietary implementations of industry accepted standards. An example is the Simple Network Management Protocol (SNMP) for IP-based networks.
- Critical transport systems, subsystems, component, and elements need to be able to monitor securely, detect changes in, and report the following:
  - Basic up/down operational status
  - Performance information
  - Operational configuration
  - Security status

- Management interfaces should be non-proprietary. They must be accessible to a wide variety of management products and solutions via open-standards-based interfaces. The interfaces should not require hard-coding to obtain operational status information about a particular system.

- To support the development of NetOps Situational Awareness capabilities, ensure that operational management solutions can share operational status and other types of management information with management solutions operated by other types of service providers. The exchange must use non-proprietary standards-based interfaces. While this could be as simple as offering a browser-accessible Web interface using HTTP or HTTPS, management product vendors are beginning to implement Web services interfaces that use SOAP to share information between management systems.

Use Accepted Industry Standards and Emerging NetOps Concepts

- Operational concepts, architectures, processes, and procedures used by transport communications and network providers must incorporate emerging NetOps concepts. They should be based on accepted industry standards.

- Take an active role in the growing NetOps community. Develop the operational policies, processes, and procedures that enhance the flow of information between different management domains. This will ensure proactive problem detection, isolation, and resolution with minimum impact on the user. [R1262]

- To support this goal, adopt and implement operational policies, processes, and procedures based on internationally accepted de facto Telecommunication Service Provider and IT Service Management (ITSM) standards.

Support Standardized DoD Service-Oriented Environment

- Employ DoD-adopted standards for implementing and using transport infrastructure in the GIG-ES Enterprise Service Management (ESM)/NetOps service-oriented environment, rather than a domain or system-oriented environment.

- A Working Group established early in CY2003 to help develop DoD-level policy for operating in a service-oriented environment is co-chaired by ASD(NII)/DoD CIO and DISA. This group has enjoyed wide participation and representation from across the Services as well as from key enterprise programs. The main focus of this group has been to formulate initial ESM/NetOps requirements for GIG-ES and for the Net-Centric Enterprise Services (NCES) Program. The group also identified DoD-level policy areas that may need to be revised to support net-centric operations in a service-oriented architecture (SOA). In addition, the group has collaborated with the NetOps CONOPS group to broaden the current transport- and network-centric approach to one that is more holistic and consistent in monitoring, managing, and controlling systems, services, and applications, in addition to transport systems and networks.

Employ DoD-Adopted Standards to Support Cross-System and Domain Management

- Employ DoD-adopted standards for operating and managing transport services. This includes interaction with counterparts in other networks or management domains, such as system or application managers.

- Specify interfaces and/or standards for the following:

  - Sharing operational status and performance information

  - Collecting and disseminating service management information

  - Selecting the format in which it is made available (e.g., SNMP, XML, CIM, SOAP)

Note: *Volume 1 of the DISR [R1179] identifies SNMP and XML as mandated standards and CIM as an emerging standard; the NCOW RM [R1179] identifies CIM as a target standard.*

Plan for Coalition Interoperability

- Plan for operations and management of transport services. This includes interacting with counterparts in other networks or management domains used by coalition partners. Most recent conflicts have involved not only U.S. forces, but forces from allies and coalition partners. In the future, U.S. information and communications

systems must support interoperability with these groups. There are various ways to achieve interoperability including the following:

- Acquisition of common systems
- Development of diverse but interoperable systems
- Adherence to standards and commercial best practices

P1307: Open Technology Development

The Deputy Under Secretary of Defense (DUSD) for Advanced Systems and Concepts (AS&C) chartered the development of the OSD ***Open Technology Development Roadmap***.[R1288] The roadmap proposes that DoD adopt generally understood OTD practices regarding open source code access, open interfaces and systems, and collaborative development methodologies. The goal is to keep pace with technology advances and changing requirements in an efficient manner.

There are five aspects associated with OTD:

- Open Architecture [P1309]
- Open Standards [P1310]
- Open Development Collaboration [P1311]
- Open Source (Software) [P1312]
- Open Systems [P1313]

P1309: Open Architecture

Open Architecture

Open Architecture (OA), according to ***Open Architecture Principles and Guidelines***,[R1288] is a pattern of nonfunctional requirements that contribute to the ability to create, deploy and manage OA systems. In some domains, e.g. systems engineering, OA considerations would apply to both hardware and software components. An Open Architecture employs open standards for key interfaces within a system [Open Systems Joint Task Force]. Open Architecture is the confluence of business and technical practices yielding modular, interoperable systems that adhere to open standards with published interfaces. This approach significantly increases opportunities for innovation and competition, enables reuse of components, facilitates rapid technology insertion, and reduces maintenance constraints. OA delivers increased warfighting capabilities in a shorter time at reduced cost [Naval Open Architecture Rhumb Lines; Open Architecture 12 Dec 06.pdf].

For an architecture to be "open" it must meet all of the following criteria.

> ***Note:*** *Specific terms are defined in Sections 2.1.2 through 2.1.7 of the* ***Open Architecture Principles and Guidelines****; links to applicable NESI Perspectives are in brackets following each question.*

- Modular
  - Is the architecture partitioned into discrete, self-contained modules of functionality?
    - [NESI on Implementing a Component-Based Architecture [P1034]]
  - Do each of the modules have well defined, published interfaces?
    - [NESI on Public Interface Design [P1060]]
    - [NESI on Standard Interface Documentation [P1069]]
    - [NESI on Key Interface Profiles (KIPs) [P1173]]
  - Are the interface definitions designed for ease of understanding by third-party architects?
    - [NESI on Exposing Functionality through Non-Standard Interfaces [P1218]]
- Interoperable
  - Do the architecture modules enable the useful exchange of data and information with other systems outside of the architecture?
    - [NESI on Net-Centric Information Engineering [P1133]]
  - Does each architecture module provide for the execution of its capabilities in response to requests coming from outside the respective module?
    - [NESI on the Software Communication Architecture (SCA) [P1087]]
    - [NESI on Services [P1164]]
    - [NESI on Phases of SOA Adoption [P1238]]
  - Does each architecture module provide for the request for execution of capabilities that are instantiated outside of the respective module?
    - [NESI on Core Enterprise Services Definitions and Status [P1166]]
  - Are architecture module interfaces based on the use of open standards?
    - [NESI on Open Standards [P1310]]
- Extensible

- Is the architecture designed with points of integration (e.g. module interfaces) that allow for future modules and capabilities to be added to the implementation, without requiring a modification to the architecture or existing implementation?
  - [NESI on Implementing Component-Based Architectures [P1034]]
- Reusable
  - Is the architecture designed with modules that can be used in multiple contexts to provide similar capabilities in those different contexts?
    - [NESI Pattern for Re-Implementation [P1220]]
    - [NESI Contracting Guidance for Reuse [P1123]]
- Composeable
  - Is the architecture comprised of modules that can be selected and assembled in various combinations to satisfy specific user requirements?
    - [NESI on Implementing a Component-Based Architecture [P1034]]
- Maintainable
  - Can the architecture's modules be maintained (revised, repaired, and replaced) without impacting the prescribed requirements (performance, availability, etc.) of the architecture's other modules?
    - [NESI on Management Issues for Exposed Functionality [P1227]]
    - [NESI on Maintaining the Internal Component Environment [P1134]]

P1310: Open Standards

The DoD Open Systems Joint Task Force defines Open Standards as standards that are widely used, consensus-based, published, and maintained by recognized standards organizations [OSJTF Terms & Definitions]. For a standard to be "open," it must meet the follow criteria:

- Is the standard widely-used?
- Is the standard consensus-based (developed using an open consortium approach)?
- Is the standard maintained and recognized by one or more recognized standards organizations, such as the Internet Society (ISOC), the Object Management Group (OMG), the Organization for the Advancement of Structured Information Standards (OASIS), or the World Wide Web Consortium (W3C)?
- Does each standard include all details necessary for interoperable implementation?
- Is the standard freely and publicly available under royalty-free terms?
- Are all patents to the implementation of the standard licensed under royalty-free terms for unrestricted use or covered by a promise of non-assertion when practiced by open source software?
- Is the standard free of all requirements for execution of a license agreement, non-disclosure agreement, grant, click-through arrangement, or any form of paperwork, to deploy conforming implementations of the standard?
- Is the standard free of all requirements for other technology that fails to meet this "open standard" criteria?

P1311: Open Development Collaboration

Open Development Collaboration is a team-based process to design, acquire, implement, deploy, and utilize a system. Include appropriately qualified subject matter experts from both government and industry, and include representatives of all stakeholders involved in the acquisition, deployment, and utilization of the system. Document the team's collaboration, correspondence, and decisions using an on-line mechanism (e.g. ,Web-based forum) that provides persistence and read/write access to all team members; the government should retain all rights to the content placed in the on-line mechanism. The government may restrict access to this content to members of the respective team, as may be deemed necessary by the government representatives. The development collaboration is "open" if it meets all of the following criteria:

- Does the collaboration cover all aspects of the development lifecycle including design, acquisition, implementation, deployment, and utilization?
- Is the team that is collaborating comprised of appropriately qualified subject matter experts from both government and industry?
- Does the team that is collaborating include representatives of all stakeholders involved in the acquisition, deployment, and utilization of the system?
- Are the team's collaboration, correspondence, and decisions persistently documented using an on-line mechanism (such as forums)?
- Is that content/documentation freely accessible to all team members?
- Do all team members have read/write access to that documentation (and is the integrity of each team member's input preserved)?
- Does the government have full rights to that content?

Examples of Open Development Collaboration

- Source Forge - example of an open development collaboration site on the Internet
- NESI Collaboration Site - example of a development collaboration site with controlled access for authorized government users, contractors, and vendors
- TBMCS DEVnet - example of a development collaboration site with controlled access for authorized government users, contractors, and vendors

P1312: Open Source (Software)

The principle of "Open Source" does not just mean access to the source code is freely and publicly available. The ***Open Source Initiative*** Open Source Definition includes ten criteria which form the basis of the following questions (note that links to applicable NESI Perspectives are in brackets after some of the questions). For software to meet the definition of "open source" it must satisfy the ten criteria.

- Is the license free of all restrictions (e.g., all royalties and other such fees for sale or use) preventing the DoD from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources?
  - [NESI Contracting Guidance for Acquisition [P1121]]
  - [NESI Contracting Guidance for Reuse [P1123]]
  - [NESI Guidance for Representations, Certifications, and other Statements of Offerors [P1126]]
- Does the program include source code and allow for distribution of that source code in textual form as well as in compiled form?
  - [NESI Guidance for Standard Interface Documentation [P1069]]
  - [NESI Guidance for RFP Section J - List of Attachments [P1125]]
- Does the license allow for modifications and derived works, and allow those changes to be distributed under the same terms as the license of the original software?
- Does the license protect the integrity of the author's original source code? For example,
  - requiring derived works to carry a different name or version number from the original software?
  - requiring that the original source code be distributed as pristine based sources plus patches, so that "unofficial" changes (those made and added to the source by parties other than the original author) can be made available but easily distinguished from the base source?
- Is the license free from all restrictions which discriminate against any person or group of persons? (External policy might place such restrictions.)
- Is the license free from all restrictions that would prevent anyone from making use of the software in a specific field or endeavor?
- Are the rights attached to the software applicable to all whom the software is redistributed without the need for execution of an additional license by those parties?
- Are the rights attached to the software free from all dependencies on the software's being part of a particular software redistribution? (If the software is extracted from that distribution and used or distributed within the terms of the software's license, all parties to whom the software is redistributed should have the same rights as those granted in conjunction with the original software distribution.)
- Is the license free from all restrictions on other software that is distributed along with the licensed software? (For example, the license must not insist that all other software distributed on the same medium must be open source software.)
- Is the license free of all provisions that may be predicated on any individual technology or style of interface? (The license must be technology-neutral.)

P1313: Open Systems

The DoD Open Systems Joint Task Force (OSJTF) defines an **open system** as "a system that employs modular design, uses widely supported and consensus based standards for its key interfaces, and has been subjected to successful validation and verification tests to ensure the openness of its key interfaces" [OSJTF What is an Open System?]. The Acquisition Community Connection, hosted by the Defense Acquisition University, has additional information concerning Modular Open Systems Approach (MOSA), the DoD "open systems" implementation [ACC Community Browser].

The Carnegie Mellon University Software Engineering Institute further defines an open system as a collection of interacting software, hardware, and human components designed to satisfy stated needs with interface specifications of its components that are fully defined, available to the public and maintained according to group consensus in which the implementations of the components conform to the interface specifications [SEI Glossary].

For a system to be considered "open" it must meet all of the following criteria:

- Is the system based on an Open Architecture?
- Does the system employ Open Standards for its **key interfaces**?
- Are the system's key interfaces maintained using an Open Development Collaboration process?
- Are the system's key interfaces fully defined and available to the public, as is the case with Open Source?

P1279: Naval Open Architecture

Interoperability, Maintainability, Extensibility, Composeability, and Reusability are non-functional requirements (NFRs) that support Open Architecture according to the ***Open Architecture Principles and Guidelines*** [R1307] which defines two types of relationships between NFRs, ***Enabled By*** and ***Facilitated By***. Enabled by is a strict dependence between NFRs while an NFR that facilitates another NFR is not required but contributes.

Below is the relationship between the NFRs

	Enabled By	***Facilitated By***
Interoperability		Open Standards
Maintainability		Composeability
		Reusability
Extensibility	Modularity	Interoperability
Composeability	Reusability	
Reusability	Interoperability	
	Extensibility	

Detailed Perspectives

- Interoperability [P1280]
- Maintainability [P1281]
- Extensibility [P1282]
- Composeability [P1283]
- Reusability [P1284]

P1280: Interoperability

Naval Open Architecture (OA) defines **interoperability** as being facilitated by **Open Standards** which makes capabilities of a system a known quantity. OA does not restrict interoperability to the use of Open Standards.

Enablers of interoperability include the following:

- Well designed and documented key internal interfaces
- Accessible metadata repository for syntactic interoperability
- **Community of Interest** (**COI**) established and standardized data models and metadata
- Availability of data
- Web service discovery
- Enterprise wide information assurance practices
- Producer and consumer decoupling through message or event-driven service bus

Inhibitors to interoperability include the following:

- Proprietary and/or unpublished APIs
- Point to point connectivity
- Application data models elevated to Enterprise data models
- Fine-grained service calls

Guidance

- G1001: Use formal standards to define public **interfaces**.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1008: Isolate platform-specific **interfaces.**
- G1011: Make components independently deployable.
- G1012: Use a set of services to expose **Component** functionality.
- G1018: Assign version identifiers to all public interfaces.
- G1035: Follow W3C standards for code which will generate a Web page display.
- G1071: Use vendor-neutral interface connections to the enterprise (e.g., **LDAP**, **JNDI**, **JMS**, databases).
- G1073: Isolate vendor extensions to enterprise-services standard interfaces.
- G1078: Document the use of non-**Java EE**-defined **deployment descriptors**.
- G1080: Adhere to the **Web Services Interoperability Organization** (**WS-I**) Basic Profile specification for **Web service** environments.
- G1085: Establish a **registered namespace** in the **XML Gallery** in the **DoD Metadata Registry** for all DoD Programs.
- G1093: Implement exception handlers for **SOAP**-based **Web services**.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.
- G1127: Use a **UDDI** specification that supports publishing discovery services.
- G1131: Use industry standard Universal Description, Discovery, and Integration (**UDDI**) **APIs** for all UDDI inquiries.
- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.
- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.

- G1209: For Java, use **JDK** logging facilities.
- G1210: For **.NET**, use Debug and Trace from the **`System.Diagnostics` namespace**.
- G1225: Use a build tool that is independent of the **Integrated Development Environment**.
- G1237: Do not **hard-code** the configuration data of a **Web service** vendor.
- G1245: Isolate the **Web service portlet** from platform dependencies using the **Web Services for Remote Portlets** (**WSRP**) Specification protocol.
- G1267: Use industry standard **HTML** data entry fields on **Web pages**.
- G1268: Label all data entry fields.
- G1270: Include scroll bars for text entry areas if the data buffer is greater than the viewable area.
- G1276: Do not modify the contents of the Web browser's status bar.
- G1277: Do not use tickers on a Web site.
- G1278: Use the browser default setting for links.
- G1284: Use only one font for **HTML** body text.
- G1285: Use **relative font sizes**.
- G1286: Provide text labels for all buttons.
- G1287: Provide feedback when a transaction will require the user to wait.
- G1292: Use text-based Web site navigation.
- G1294: Provide a site map on all Web sites.
- G1295: Provide redundant text links for images within an **HTML** page.
- G1300: Secure all **endpoints**.
- G1301: Practice layered security.
- G1302: Validate all inputs.
- G1304: Unit test all code.
- G1306: **Identify** and **authenticate** users of the application.
- G1308: Configure **Public Key Enabled** applications to use a **Federal Information Processing Standard** (**FIPS**) 140-2 certified cryptographic module.
- G1309: Make applications handling high value unclassified information in Minimally Protected environments **Public Key Enabled** to interoperate with **DoD High Assurance** .
- G1310: Protect application cryptographic objects and functions from tampering.
- G1311: Use **Hypertext Transfer Protocol over Secure Socket Layer** (**HTTPS**) when applications communicate with DoD **Public Key Infrastructure** (**PKI**) components.
- G1312: Make applications capable of being configured for use with DoD **PKI**.
- G1314: Provide applications the ability to import and export keys (software certificates only).
- G1316: Ensure that applications protect **private keys**.
- G1317: Ensure applications store **Certificates** for subscribers (the owner of the **Public Key** contained in the Certificate) when used in the context of signed and/or encrypted email.
- G1318: Develop applications such that they provide the capability to manage and store **trust points** (**Certificate Authority** Public Key **Certificates**).
- G1319: Ensure applications can recover data encrypted with legacy keys provided by the DoD **PKI** Key Recovery Manager (**KRM**).
- G1320: Use a minimum of 128 bits for **symmetric keys**.
- G1321: Enable applications to be capable of performing **Public Key** operations necessary to verify signatures on DoD **PKI** signed objects.
- G1322: Ensure that applications that interact with the DoD **PKI** using **SSL** (i.e., **HTTPS**) are capable of performing cryptologic operations using the **Triple Data Encryption Algorithm** (**TDEA**).

- G1323: Generate random **symmetric encryption** keys when using symmetric encryption.
- G1324: Protect **symmetric keys** for the life of their use.
- G1325: Encrypt **symmetric keys** when not in use.
- G1326: Ensure applications are capable of producing Secure Hash Algorithm (**SHA**) **digests** of **messages** to support verification of DoD **PKI** signed objects.
- G1327: Enable an application to obtain new **Certificates** for subscribers.
- G1328: Enable an application to retrieve **Certificates** for use, including relying party operations.
- G1330: Ensure applications are capable of checking the status of **Certificates** using a **Certificate Revocation List** (**CRL**) if not able to use the **Online Certificate Status Protocol** (**OCSP**).
- G1331: Ensure applications are able to check the status of a Certificate using the **Online Certificate Status Protocol** (**OCSP**).
- G1333: Only use a **Certificate** during the Certificate's validity range, as bounded by the Certificate's "Validity - Not Before" and "Validity - Not After" date fields.
- G1335: Make applications capable of being configured to operate only with PKI Certificate Authorities specifically approved by the application's owner/managing entity.
- G1338: Ensure that **Public Key** Enabled applications support multiple organizational units.
- G1339: Practice defensive programming by checking all method arguments.
- G1341: Use a security manager support to restrict application access to privileged resources.
- G1343: Declare classes final to stop inheritance and prevent methods from being overridden.
- G1344: Encrypt sensitive data stored in configuration or resource files.
- G1347: Secure remote connections to a database.
- G1349: Validate all input that will be part of any dynamically generated **SQL**.
- G1350: Implement a strong password policy for **RDBMS**.
- G1351: Enhance database security by using multiple user accounts with constraints.
- G1352: Use database clustering and redundant array of independent disks (RAID) for high availability of data.
- G1356: Use the **SOAP** standard for all **Web services**.
- G1357: Do not rely solely on transport level security like **SSL** or **TLS**.
- G1359: Bind **SOAP Web service** security policy assertions to the service by expressing them in the associated **WSDL** file.
- G1362: Validate XML messages against a **schema**.
- G1363: Do not use clear text passwords.
- G1364: Hash all passwords using the combination of a timestamp, a **nonce** and the password for each **message** transmission.
- G1365: Specify an expiration value for all security tokens.
- G1366: Digitally sign all **messages** where non-repudiation is required.
- G1367: Digitally sign **message** fragments that are required not to change during transport.
- G1369: Digitally sign all requests made to a security token service.
- G1371: Use the **Digital Signature Standard** for creating **Digital Signatures**.
- G1372: Use an X.509 **Certificate** to pass a **Public Key**.
- G1373: **Encrypt messages** that cross an **IA** boundary.
- G1374: Individually **encrypt** sensitive **message** fragments intended for different intermediaries.
- G1376: Do not **encrypt** message fragments that are required for correct **SOAP** processing.
- G1377: Use **LDAP** 3.0 or later to perform all connections to LDAP repositories.
- G1378: Encrypt communication with **LDAP** repositories.

- G1379: Use **SAML** version 2.0 for representing security assertions.
- G1380: Use the **XACML** 2.0 standard for **SAML**-based rule engines.
- G1381: Encrypt sensitive persistent data.
- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1569: Maintain a comprehensive list of all of the **Components** that are part of the Node.
- G1570: Assume an active management role among the **Components** within the Node.
- G1581: Expose legacy functionality through the use of a service.
- G1635: Make Nodes that will be part of the **Global Information Grid** (**GIG**) consistent with the *GIG Integrated Architecture*.
- G1636: Comply with the **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**).
- G1637: Make Node-implemented **directory services** comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**).
- G1638: Comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node directory services **proxies**.
- G1640: Register **Components** exposed by the Node with the **DISA**-hosted registries.
- G1641: Comply with the Service Discovery **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node-implemented **Service Discovery** (**SD**).
- G1642: Comply with the **Service Discovery Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node **Service Discovery** (**SD**) **proxies**.
- G1644: Comply with the **Federated Search** - **Search Web Service** (**SWS**) **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node implemented Federated Search - Search Web Service (SWS).
- G1645: Implement a local **Content Discovery Service** (**CDS**).
- G1646: Comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles (KIPs**) in Node **Federated Search** Services **proxies**.
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.
- G1724: Develop XML documents to be well formed.
- G1725: Develop XML documents to be **valid** XML.
- G1726: Define XML Schemas using **XML Schema Definition** (XSD).
- G1727: Provide names for XML type definitions.
- G1728: Define types for all **XML elements**.
- G1730: Follow an XML coding standard for defining schemas.
- G1737: Define a target namespace in schemas.
- G1746: Develop XSLT **style sheets** that are XSLT version agnostic.

- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1754: Give each new XML schema version a unique URL.
- G1759: Use a style guide when developing Web portlets.
- G1761: Provide units of measurements when displaying data.
- G1763: Indicate the security classification for all classified data.
- G1770: Explicitly define **Data Distribution Service** (DDS) **Domains**.
- G1771: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of a **publisher**.
- G1772: Assign a unique identifier for each **Data-Distribution Service** (DDS) **Domain**.
- G1785: Stipulate that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on reuse of common functionality.
- G1786: Stipulate that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on well defined services.
- G1787: Stipulate that the Offeror is to use the NESI *Net-Centric Implementation* documentation set to assess net-centric interoperability.
- G1796: Explicitly define **Data Distribution Service** (DDS) **Domain Topics**.
- G1798: Explicitly define all the **Data Distribution Service** (DDS) **Domain data types**.
- G1799: Explicitly associate data types to the **Data Distribution Service** (DDS) **Topics** within a DDS **Domain**
- G1800: Explicitly identify Keys within the **Data Distribution Service** (DDS) **data type** that uniquely identify an instance of a data object.
- G1801: Explicitly define a **Topic Quality of Service** (QoS) for each **Data Distribution Service** (DDS) Topic within a DDS **Domain**.
- G1803: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe real-time messaging criteria for **Publishers**.
- G1804: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe **DataWriter**.
- G1805: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of the **Subscriber**.
- G1806: Explicitly define the Request-Offered **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of the **DataReader**.
- G1808: Handle all **Data Distribution Service** (DDS) **Quality of Service** (QoS) contract violations using one of the **Subscriber access APIs**.
- G1810: Use **data models** to document the data contained within the **Data Distribution Service** (DDS) **Data-Centric Publish Subscribe** (DCPS).
- G1797: Use a minimum of 1024 bits for **asymmetric keys**.
- G1090: Do not **hard-code** a **Web service's endpoint**.

Best Practices

- BP1007: Develop software using **open standard Application Programming Interfaces** (**APIs**).
- BP1392: Register services in accordance with a documented service registration plan.

P1281: Maintainability

In the Naval Open Architecture (OA) context, maintainability is "the portion of a component's or system's lifecycle after installation, including its end of life. Key to this lifecycle is updating the system to introduce new technology, changed business processes, etc." (see ***Open Architecture Principles and Guidelines*** section 2.1.7.1 [R1307]). Maintainability depends on a modular system with well-defined interfaces and documentation for all aspects of the lifecycle of a system.

Enablers of maintainability include the following:

- Modular design with well-defined, stable interfaces
- Loose coupling
- Clear and concise documentation
- Use cases and testing
- Compliance with open standards

Inhibitors of maintainability include the following:

- Frequent changes to interfaces
- Tightly coupled and heavily optimized solutions

Guidance

- G1001: Use formal standards to define public **interfaces**.
- G1002: Separate public **interfaces** from implementation.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1004: Make public **interfaces** backward-compatible within the constraints of a published **deprecation** policy.
- G1018: Assign version identifiers to all public interfaces.
- G1019: Deprecate public interfaces in accordance with a published deprecation policy.
- G1022: Insulate public **interfaces** from compile-time dependencies.
- G1027: Internally document all source code developed with DoD funding.
- G1032: Validate all input fields.
- G1043: Separate formatting from data through the use of **style sheets** instead of hard coded **HTML** attributes.
- G1044: Comply with Federal accessibility standards contained in Section 508 of the Rehabilitation Act of 1973 (as amended) when developing software user interfaces.
- G1052: Use the code-behind feature in ASP.NET to separate presentation code from the business logic.
- G1053: Do not embed HTML code in any code-behind code used by aspx pages.
- G1056: Specify a versioning policy for **.NET** assemblies.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.
- G1060: Encapsulate Java code that is used in **JSP**(s) in tag libraries.
- G1071: Use vendor-neutral interface connections to the enterprise (e.g., **LDAP**, **JNDI**, **JMS**, databases).
- G1073: Isolate vendor extensions to enterprise-services standard interfaces.
- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).
- G1083: Do not pass **Web Services-Interoperability Organization** (**WS-I**) **Document Object Model** (**DOM**) documents as strings.
- G1085: Establish a **registered namespace** in the **XML Gallery** in the **DoD Metadata Registry** for all DoD Programs.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.

- G1090: Do not **hard-code** a **Web service's endpoint**.
- G1094: Catch all exceptions for application code exposed as a **Web service**.
- G1095: Use **W3C** fault codes for all **SOAP** faults.
- G1118: Localize **CORBA** vendor-specific source code into separate **modules**.
- G1121: Do not modify **CORBA** Interface Definition Language (**IDL**) compiler auto-generated stubs and skeletons.
- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.
- G1146: Include information in the **data model** necessary to generate a **data dictionary**.
- G1147: Use **domain analysis** to define the constraints on input data validation.
- G1148: **Normalize** data models.
- G1151: Define declarative **foreign keys** for all relationships between tables to enforce **referential integrity**.
- G1153: Separate application, presentation, and data tiers.
- G1154: Use **stored procedures** for operations that are focused on the insertion and maintenance of data.
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.
- G1204: Create configuration services to provide distributed user control of the appropriate configuration parameters.
- G1205: Use non-source code persistence to store all user-modifiable **CORBA** service configuration parameters.
- G1208: Add new functionality rather than redefining existing interfaces in a manner that brings incompatibility.
- G1213: Provide an architecture design document.
- G1214: Provide a document with a plan for **deprecating** obsolete **interfaces**.
- G1215: Provide a coding standards document.
- G1216: Provide a software release plan document.
- G1217: Develop and use externally configurable components.
- G1218: Use a build tool that supports operation in an automated mode.
- G1219: Use a build tool that checks out files from configuration control.
- G1220: Use a build tool that **compiles** source code and dependencies that have been modified.
- G1221: Use a build tool that creates libraries or archives after all required compilations are completed.
- G1222: Use a build tool that creates executables.
- G1223: Use a build tool that is capable of running unit tests.
- G1224: Use a build tool that cleans out intermediate files that can be regenerated.
- G1225: Use a build tool that is independent of the **Integrated Development Environment**.
- G1237: Do not **hard-code** the configuration data of a **Web service** vendor.
- G1239: Use **design patterns** (e.g., **facade**, **proxy**, or **adapter**) or property files to isolate vendor-specifics of vendor-dependent connections to the enterprise.
- G1267: Use industry standard **HTML** data entry fields on **Web pages**.
- G1271: Provide instructions and **HTML** examples for all style sheets.
- G1283: Use **linked style sheets** rather than embedded styles.
- G1300: Secure all **endpoints**.
- G1301: Practice layered security.
- G1308: Configure **Public Key Enabled** applications to use a **Federal Information Processing Standard** (**FIPS**) 140-2 certified cryptographic module.
- G1309: Make applications handling high value unclassified information in Minimally Protected environments **Public Key Enabled** to interoperate with **DoD High Assurance** .

- G1311: Use **Hypertext Transfer Protocol over Secure Socket Layer** (**HTTPS**) when applications communicate with DoD **Public Key Infrastructure** (**PKI**) components.
- G1312: Make applications capable of being configured for use with DoD **PKI**.
- G1313: Provide documentation for application configuration for use with DoD **PKI**.
- G1314: Provide applications the ability to import and export keys (software certificates only).
- G1318: Develop applications such that they provide the capability to manage and store **trust points** (**Certificate Authority** Public Key **Certificates**).
- G1319: Ensure applications can recover data encrypted with legacy keys provided by the DoD **PKI** Key Recovery Manager (**KRM**).
- G1320: Use a minimum of 128 bits for **symmetric keys**.
- G1321: Enable applications to be capable of performing **Public Key** operations necessary to verify signatures on DoD **PKI** signed objects.
- G1322: Ensure that applications that interact with the DoD **PKI** using **SSL** (i.e., **HTTPS**) are capable of performing cryptologic operations using the **Triple Data Encryption Algorithm** (**TDEA**).
- G1323: Generate random **symmetric encryption** keys when using symmetric encryption.
- G1324: Protect **symmetric keys** for the life of their use.
- G1325: Encrypt **symmetric keys** when not in use.
- G1326: Ensure applications are capable of producing Secure Hash Algorithm (**SHA**) **digests** of **messages** to support verification of DoD **PKI** signed objects.
- G1327: Enable an application to obtain new **Certificates** for subscribers.
- G1328: Enable an application to retrieve **Certificates** for use, including relying party operations.
- G1330: Ensure applications are capable of checking the status of **Certificates** using a **Certificate Revocation List** (**CRL**) if not able to use the **Online Certificate Status Protocol** (**OCSP**).
- G1331: Ensure applications are able to check the status of a Certificate using the **Online Certificate Status Protocol** (**OCSP**).
- G1333: Only use a **Certificate** during the Certificate's validity range, as bounded by the Certificate's "Validity - Not Before" and "Validity - Not After" date fields.
- G1335: Make applications capable of being configured to operate only with PKI Certificate Authorities specifically approved by the application's owner/managing entity.
- G1338: Ensure that **Public Key** Enabled applications support multiple organizational units.
- G1340: Log all exceptional conditions.
- G1342: Restrict direct access to class internal variables to functions or methods of the class itself.
- G1343: Declare classes final to stop inheritance and prevent methods from being overridden.
- G1346: Audit database access.
- G1348: Log database **transactions**.
- G1352: Use database clustering and redundant array of independent disks (RAID) for high availability of data.
- G1356: Use the **SOAP** standard for all **Web services**.
- G1359: Bind **SOAP Web service** security policy assertions to the service by expressing them in the associated **WSDL** file.
- G1372: Use an X.509 **Certificate** to pass a **Public Key**.
- G1378: Encrypt communication with **LDAP** repositories.
- G1576: Provide an environment to support the development, build, integration, and test of net-centric capabilities.
- G1577: Maintain an **Enterprise Service** schedule for interim and final **enterprise** capabilities within the Node.
- G1578: Define a schedule for **Components** that includes the use of the **Enterprise Services** defined within the Node's enterprise service schedule.

- G1582: In Node **Enterprise Service** schedules, include version numbers of standard Enterprise Services interfaces being implemented.
- G1583: Provide routine **Enterprise Services** schedule updates to every **component** of a Node.
- G1717: Use constants instead of hard-coded numbers for characteristics that may change throughout the lifetime of the model.
- G1718: Design circuits to be synchronous.
- G1719: Automate testbench error checking in VHDL development.
- G1727: Provide names for XML type definitions.
- G1728: Define types for all **XML elements**.
- G1729: Annotate XML type definitions.
- G1730: Follow an XML coding standard for defining schemas.
- G1731: Only reference **XML elements** defined by a Type in substitution groups.
- G1735: Use the **`.xsd`** file extension for files that contain XML Schema definitions.
- G1736: Separate document schema definition and document instance into separate documents.
- G1740: Append the suffix Type to XML type names.
- G1744: Only reference abstract **XML elements** in substitution groups.
- G1745: Append the suffix Group to substitution group **XML element** names.
- G1751: Document all XSLT code.
- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1754: Give each new XML schema version a unique URL.
- G1755: Use accepted file extensions for all files that contain XSL code.
- G1756: Isolate XPath expression statements into the configuration data.
- G1773: Use **`#include`** guards for all headers.
- G1774: Make header files self-sufficient.
- G1775: Do not overload the logical **`AND`** operator.
- G1776: Do not overload the logical **`OR`** operator.
- G1777: Do not overload the **`comma`** operator.
- G1778: Place all **`#include`** statements before all namespace **`using`** statements.
- G1779: Explicitly namespace-qualify all names in header files.

Best Practices

- BP1021: Create fully encapsulated classes.

P1282: Extensibility

Extensible systems facilitate adding future capabilities and points of contact or integration. To support this, Open Architecture defines an extensible system as one with "sufficient internal quality and compartmentalization of data and behavior that new capabilities do not introduce unintended changes to existing data and behavior" (see ***Open Architecture Principles and Guidelines*** [R1307]). To achieve this, a system must be modular and interoperable.

Enablers of extensibility include the following:

- Well defined points of variability
- Layered architecture
- Loose coupling

Inhibitors to extensibility include the following:

- Undocumented design and architecture assumptions

Guidance

- G1002: Separate public **interfaces** from implementation.
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.
- G1271: Provide instructions and **HTML** examples for all style sheets.

P1283: Composeability

Composeable systems allow for components to be selected and assembled in different ways to meet user requirements. In order for a system to be composeable its components must also be reusable, interoperable, extensible, and modular as defined by Open Architecture.[R1307]

Enablers of composeability include the following:

- Standard enterprise ontology
- Enterprise service bus
- Clearly defined quality of service (QoS)
- Tools for composing services

Inhibitors to composeability include the following:

- No enterprise architecture management

Guidance

- G1002: Separate public **interfaces** from implementation.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1011: Make components independently deployable.
- G1012: Use a set of services to expose **Component** functionality.
- G1022: Insulate public **interfaces** from compile-time dependencies.
- G1045: Define **XML** format information separately in **XSL**.
- G1050: In **ASP**, isolate the presentation tier from the middle tier using **COM** objects.
- G1052: Use the code-behind feature in ASP.NET to separate presentation code from the business logic.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.
- G1060: Encapsulate Java code that is used in **JSP**(s) in tag libraries.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1144: Develop two-level database models: one level captures the **conceptual** or logical aspects, and the other level captures the **physical** aspects.
- G1153: Separate application, presentation, and data tiers.
- G1155: Use **triggers** to enforce **referential** or data integrity, not to perform complex **business logic**.
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.
- G1719: Automate testbench error checking in VHDL development.

P1284: Reusability

Open Architecture defines a reusable artifact as one that provides a capability that can be used in multiple contexts. Reuse is not confined to a software component but any lifecycle artifact including training, documentation, and configuration. Open Architecture is concerned with artifacts which relate to the design, construction, and configuration of a component.

Enablers of reusability include the following:

- Use of Reusable Asset Specification (RAS)
- Low code complexity
- Components that depend primarily on OA interfaces

Inhibitors to reusability include the following:

- Serialized or single-threaded implementation
- Proprietary standards
- Cut-and-paste programming

Guidance

- G1019: Deprecate public interfaces in accordance with a published deprecation policy.
- G1045: Define **XML** format information separately in **XSL**.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.
- G1060: Encapsulate Java code that is used in **JSP**(s) in tag libraries.
- G1144: Develop two-level database models: one level captures the **conceptual** or logical aspects, and the other level captures the **physical** aspects.
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.
- G1217: Develop and use externally configurable components.
- G1271: Provide instructions and **HTML** examples for all style sheets.
- G1283: Use **linked style sheets** rather than embedded styles.
- G1311: Use **Hypertext Transfer Protocol over Secure Socket Layer** (**HTTPS**) when applications communicate with DoD **Public Key Infrastructure** (**PKI**) components.
- G1321: Enable applications to be capable of performing **Public Key** operations necessary to verify signatures on DoD **PKI** signed objects.
- G1335: Make applications capable of being configured to operate only with PKI Certificate Authorities specifically approved by the application's owner/managing entity.
- G1356: Use the **SOAP** standard for all **Web services**.
- G1377: Use **LDAP** 3.0 or later to perform all connections to LDAP repositories.
- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.

- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1569: Maintain a comprehensive list of all of the **Components** that are part of the Node.
- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.
- G1717: Use constants instead of hard-coded numbers for characteristics that may change throughout the lifetime of the model.
- G1718: Design circuits to be synchronous.
- G1719: Automate testbench error checking in VHDL development.
- G1759: Use a style guide when developing Web portlets.
- G1773: Use **`#include`** guards for all headers.
- G1774: Make header files self-sufficient.
- G1775: Do not overload the logical **`AND`** operator.
- G1776: Do not overload the logical **`OR`** operator.
- G1777: Do not overload the **`comma`** operator.
- G1778: Place all **`#include`** statements before all namespace **`using`** statements.
- G1779: Explicitly namespace-qualify all names in header files.
- G1784: Include a statement in the solicitation for Contractors to identify and list data rights for all proposed products.
- G1785: Stipulate that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on reuse of common functionality.
- G1786: Stipulate that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on well defined services.
- G1787: Stipulate that the Offeror is to use the NESI *Net-Centric Implementation* documentation set to assess net-centric interoperability.
- G1788: Stipulate that the Offeror is to use Government approved data rights labels and markings for all deliverables that are identified as Unlimited or Government Purpose Rights.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.

P1122: Relationship with the JCIDS Process

The appropriate timeframe to start implementing net-centricity and interoperability is during the early definition of the system with the preparation of the Capabilities Documents. These documents, prepared under the **Joint Capabilities Integration and Development System** (**JCIDS**), set the stage for the subsequent acquisition process. Before initiating a program, the JCIDS process identifies warfighting capability and supportability gaps and the **Doctrine, Organization, Training, Materiel, Leadership and education, Personnel, and Facilities** (**DOTMLPF**) capabilities required to fill those gaps. The documentation developed during the JCIDS process provides the formal communication of capability needs between the warfighter, acquisition, and resource management communities.

Program sponsors, in coordination with program managers, should consider applicable NESI guidance when preparing JCIDS documents. Program sponsors and managers can use Part 1 [P1286] and Part 2 [P1288] to develop a high-level foundational understanding of the relevant issues and have a starting point for planning relevant activities and strategies. Incorporating this guidance facilitates meeting the requirements of the ASD(NII) Net-Centric Checklist (see P1239 [P1239]). This is a means of increasing interoperability and aiding the development of architectural products. Program personnel should look for the attributes in the program capabilities documents (with reference to the relevant portions of NESI) that are contained in Table 1 below.

Table 1 - Relationship between JCIDS Documents, Process Milestones, and NESI Guidance

JCIDS Document	Milestones	Description	Relevant NESI Guidance
Initial Capabilities Document (ICD)	A, B, C	Defines capability gap in terms of functional area(s), relevant range of military operations, time, obstacles to overcome, and key attributes, with appropriate measures of effectiveness. Recommends materiel approach(s) based on cost analysis, efficacy, sustainability, environmental quality impacts, and associated risks.	Parts 1, 2
Capability Development Document (CDD)	B	Provides operational performance attributes, including supportability, for the acquisition community to design the proposed system. Includes key performance parameters (KPP) and other parameters that guide the development, demonstration, and testing of the current increment. Outlines the overall strategy for developing full capability.	Parts 2, 3, 4 Net-Ready Key Performance Parameter (NR-KPP) developed for this CDD
Capability Production Document (CPD)	C	Addresses the production attributes and quantities specific to a single increment of an acquisition program. Supersedes threshold and objective performance values of the CDD.	Parts 3, 4, 5 Updated NR-KPP required in this CPD

The Net-Ready Key Performance Parameter (NR-KPP) noted in Table 1 measures the net-centricity of a new program or major upgrade. The NR-KPP contains four elements:

- Compliance with the **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**)
- Compliance with applicable Global Information Grid **Key Interface Profiles** (**KIPs**)
- Compliance with DoD **information assurance** (**IA**) requirements
- Support for integrated architecture products that assess information exchange and use for a given capability

Refer to the **Defense Acquisition University** (**DAU**) Defense Acquisition Guidebook Section 7.3.4 for further information on the NR-KPP elements.

The program sponsor and manager can also use NESI to aid in the development of the NR-KPP as show in Table 2.

Table 2 - Relationship between NESI and the NR-KPP

NESI	NCOW RM Services Strategy	NCOW RM Data Strategy	NCOW RM IA Strategy	Information Assurance	Key Interface Profiles (KIPs)	Integrated Architectures
Part 1	3.2, 3.3.2, 4.4	3.2, 3.4, 4.2	3.2		3.3.1	1.5, 4.3 - 4.6
Part 2	4.1, 4.7, 7.0, 8.0	3.1 - 3.6, 8.0	5.1 - 5.7, 8.0	5.1 - 5.7, 8.0	4.1	4.1, 4.2, 6.3
Part 3	All	Net-Centric Data Strategy (NCDS)	Migration Concern: Security			Migration Concern: Architecture Documentation Maintenance, Migration Planning Process
Part 4	2.2 - 2.4	2.2 - 2.4	2.2 - 2.4	2.2 - 2.4	2.2 - 2.4	All of Part 4, but especially 2.4 .1
Part 5	Web Services, Browser-Based Clients	Data Tier, Data, Metadata	Application Security	Application Security		Technical Guidance and Tactics
Part 6	N/A	N/A	N/A	N/A	N/A	N/A

P1362: DISR Service Areas

Programs use the **Defense IT Standards Registry** (**DISR**) Service Areas to develop DISR Online program-specific profiles. Standards and specifications registered in DISR are grouped into Service Areas, simplifying how programs profile themselves. This NESI perspective and the linked detailed perspectives provide traceability between NESI content and the DISR Service Areas. Programs can use the appropriate NESI perspectives traced to the Service Areas in the program's DISR profile to determine applicable NESI guidance.

> ***Note:*** *NESI content is not applicable to all DISR Service Areas. Thus, only those areas that both DISR and NESI cover have links to NESI perspectives (with the same names) in the following DISR Service Area list.*

Service Area List

- Aviation: Air Traffic Management
- Business Processing
- C4ISR: Payload Platform [P1363]
- Communications Applications [P1364]
- Data Interchange Services [P1365]
- Data Management Services [P1366]
- Devices (Smart Cards)
- Distributed Computing Services [P1367]
- Engineering Support
- Environment Management [P1368]
- Graphic Services
- Identification Friend of Foe
- Internationalization Services [P1369]
- Medical Services
- Multimedia
- Operating Systems Services [P1370]
- Platform Communications Services
- Security Services [P1371]
- Software Engineering Services
- System Management Services
- User Interface Services [P1372]
- User (Physical/Cognitive) [P1373]

P1363: C4ISR: Payload Platform

This service area addresses interoperability requirements for integration of C4ISR payloads like sensor packages and communications relays. This service area relates to NESI only generally through interface design, documentation, and insulation. Use the following detailed perspectives for guidance related to this service area.

Detailed Perspectives

- Standard Interface Documentation [P1069]
- Implement a Component-Based Architecture [P1034]
- Public Interface Design [P1060]

P1069: Standard Interface Documentation

This section provides guidance for documenting source code. The references provide links on documenting code for the Java and the Microsoft .NET environments. For all other languages, configuration files, and XML files, please follow the associated language-specified format for documentation.

Javadoc commands
The **Javadoc** tool parses special tags when they are embedded within a Javadoc comment. These doc tags enable a programmer to autogenerate a complete, well-formatted API from the source code. The tags start with an ampersand (@) and are case-sensitive; an "a" is different from an "A."

A tag must start at the beginning of a line, after any leading spaces and an optional asterisk, or it will be treated as normal text. By convention, group tags with the same name together. For example, put all **@see** tags together.

Guidance

- G1027: Internally document all source code developed with DoD funding.

Examples

Sample Java code with Javadoc

This is a sample Enterprise Java Bean with Javadoc tags for the API that implements a method to set a string to "Hello." Use this example to generate documents from the command line and from Ant.

```
package com.testejb;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
/**
 * This session bean demonstrates a simple session bean
 */
public class TestSessionBean implements SessionBean {
   private String test = "hello from the test ejb";
   public TestSessionBean( ){ }
   public void setSessionContext(SessionContext sc){ }
   public void ejbActivate( ){ }
   public void ejbPassivate( ){ }
   public void ejbRemove( ){ }
   public void ejbCreate( ){ }
   /**
    * This method returns the test string
    * @return the value of test
    */
   public String getTest( ) {
      return test;
   } // End getTest
   /**
    * This method sets the test string
    * @param String t
    */
   public void setTest(String t) {
      test = t;
   } // End setTest
} // End TestSessionBean
package com.testejb;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
/**
 * This session bean demonstrates a simple session bean
 */
public class TestSessionBean implements SessionBean {
   private String test = "hello from the test ejb";
   public TestSessionBean( ){ }
   public void setSessionContext(SessionContext sc){ }
   public void ejbActivate( ){ }
   public void ejbPassivate( ){ }
   public void ejbRemove( ){ }
```

```
   public void ejbCreate( ){ }
   /**
    * This method returns the test string
    * @return the value of test
    */
   public String getTest( ) {
     return test;
   } // end getTest
   /**
    * This method sets the test string
    * @param String t
    */
   public void setTest(String t) {
     test = t;
   } // End setTest
} // End TestSessionBean
```

Sample C# code with documentation tags

This sample .NET application shows the necessary comment structure to generate the interface documentation.

```
using System;
namespace HelloWorldNamespace {
   ///
   /// Hello World Example C# application
   ///
   class HelloWorldClass {
     ///
     /// The main entry point for the application.
     ///
     [STAThread]
     static void Main(string[] args) {
       // Loop through some indices and display the value
       // from GetHelloText(...)
       for ( int expressionCounter = -1; expressionCounter < 4; expressionCounter ++ ) {
         Console.Out.WriteLine (expressionCounter.ToString("#0") + ": " +
GetHelloText(expressionCounter) );
       } // End for
       Console.In.Read(); // Pause the console
     } // End main
     ///
     /// Gets a "hello" string given an index
     ///
     ///
     /// Index of the "hello" string to retrieve
     ///
     ///
     /// A "hello"string if the index is valid, otherwise
     /// an error
     ///
     static stringGetHelloText(int index) {
       string[] helloExpressions = new string[] {
         "Hello World", "Hello All", "Howdy"
       };
       if (index < 0 || index >=helloExpressions.Length) {
         return "Error";
       } // End if
       else {
         returnhelloExpressions [index];
       } // End else
     } // End get Hello
   } // EndHelloWorldClass
} // End HelloWorldNamespace
```

P1034: Implement a Component-Based Architecture

The Federation of Government Information Processing Councils/Industry Advisory Council (FGIPC/IAC) defined **component-based architecture** (**CBA**) as follows in a March 2003 paper titled Succeeding with "Component-Based Architecture in e-Government":

"An architecture process that enables the design of enterprise solutions using pre-manufactured components. The focus of the architecture may be a specific project or the entire enterprise. This architecture provides a plan of what needs to be built and an overview of what has been built already." [Succeeding with Component-Based Architecture]

CBA represents a shift from the traditional, custom-development-oriented, "design, code, and test" approach that has been used throughout the DoD in the past to a more business-oriented "architect, acquire, and assemble" approach.

The custom-development approach has been successful in building many systems. However, the integration, evolution, reuse and cost of these systems have presented a problem. Consequently, these custom-developed systems have been labeled as archaic **stovepipes** that can not plug-and-play with other systems.

CBA promises benefits such as shorter time to market, lower risk, and modular and adaptive systems.

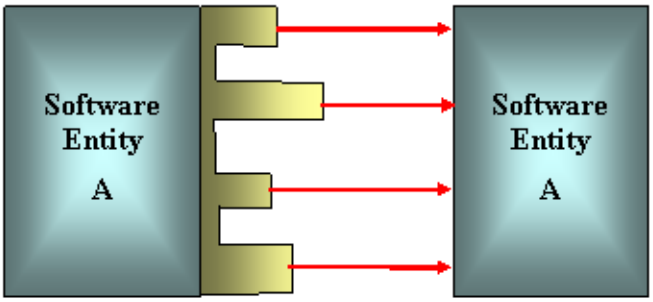
The core of CBA is components. The NESI definition of the term **component** is that it is one of the parts that make up a system; a component may be hardware or software and may be subdivided into other components. The following guidance statements capture the essence of components.

Guidance

- G1011: Make components independently deployable.
- G1012: Use a set of services to expose **Component** functionality.
- G1217: Develop and use externally configurable components.

P1060: Public Interface Design

A public interface is the logical point at which independent software entities interact. The entities may interact with each other within a single computer, across a network, or across a variety of other topologies. It is important that public **interfaces** be stable and designed to support future changes, enhancements, and **deprecation** in order for the interaction to continue.



I1007

Guidance

- G1213: Provide an architecture design document.
- G1215: Provide a coding standards document.
- G1216: Provide a software release plan document.
- G1214: Provide a document with a plan for **deprecating** obsolete **interfaces**.
- G1022: Insulate public **interfaces** from compile-time dependencies.
- G1208: Add new functionality rather than redefining existing interfaces in a manner that brings incompatibility.
- G1004: Make public **interfaces** backward-compatible within the constraints of a published **deprecation** policy.
- G1018: Assign version identifiers to all public interfaces.
- G1019: Deprecate public interfaces in accordance with a published deprecation policy.
- G1001: Use formal standards to define public **interfaces**.
- G1002: Separate public **interfaces** from implementation.
- G1003: Separate shared **Application Programming Interfaces** (**APIs**) from internal APIs.
- G1008: Isolate platform-specific **interfaces.**
- G1010: Use **open-standard** logging frameworks.
- G1073: Isolate vendor extensions to enterprise-services standard interfaces.

Best Practices

- BP1240: Present complete and coherent sets of concepts to the user.
- BP1241: Design statically typed **interfaces**.
- BP1242: Minimize an **interface's** dependencies on other **interfaces**.
- BP1243: Express **interfaces** in terms of application-level types.
- BP1244: Use assertions only to aid development and **integration**.
- BP1021: Create fully encapsulated classes.
- BP1007: Develop software using **open standard Application Programming Interfaces** (**APIs**).

P1364: Communications Applications

This service area relates to the capability to send, receive, forward, and manage electronic and voice messages. Applications include the following:

- Broadcast
- Communications conferencing
- Enhanced telephony
- Organizational messaging
- Personal messaging
- Shared-screen teleconferencing
- Video teleconferencing

NESI supportions the Communications Applications service area through guidance related to networks and transport. Use the following detailed perspectives for guidance related to this service area.

Detailed Perspectives

- Software Communications Architecture [P1087]
- Network Information Assurance [P1147]
- Node Transport [P1138]
- Text Conferencing [P1388]

P1087: Software Communication Architecture

The **Software Communications Architecture** (**SCA**) establishes an implementation-independent framework with baseline requirements for the development of software for an established hardware platform, such as software defined radios. The SCA is an architectural framework that was created to maximize portability, interoperability, and configurability of the software while still allowing the flexibility to address domain specific requirements and restrictions. Constraints on software development imposed by the framework are on the interfaces and the structure of the software and not on the implementation of the functions that are performed.

The framework places an emphasis on areas where reusability is affected and allows implementation unique requirements to determine a specific application of the architecture. SCA specifications incorporate accepted industry standards such as a subset of the **Portable Operating System Interface** (**POSIX**) specification and the **Object Management Group** (**OMG**) **CORBA** specification.

SCA includes a real-time operating system functionality to provide multi-threaded support for all software executing on the system. Software can include SCA applications, devices, and services. The exact functionality supported by the **Operating Environment** is described by the **Application Environment Profile** (**AEP**) which is a subset of the POSIX specification.

The OMG Domain Special Interest Group for Software Radios (SWRADIO DSIG) and Software Defined Radio Forum (SDRF) are working together towards building an international commercial standard based on the SCA.

The purpose of this perspective is to provide guidance and reference material for Programs providing products and services using SCA in order to increase interoperability and net-centricity.

Guidance

- G1713: Use an **Operating Environment** (**OE**) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.
- G1714: Develop **Software Communications Architecture** (**SCA**) applications to use only **Operating Environment** functionality defined by the SCA Application Environment Profile.

Best Practices

- BP1715: Design SCA log services according to the OMG Lightweight Log Service Specification.
- BP1716: Develop applications for SCA-compliant systems using a standard higher order language.
- BP1880: Justify, document, and obtain a waiver for all radio terminal acquisitions that are not JTRS/SCA compliant.

P1147: Network Information Assurance

Implementation of the DoD **Information Assurance** (IA) Strategic Plan is required to comply with the DoD **Net-Ready Key Performance Parameter** (NR-KPP). Components that implement IA, however, can be a barrier to interoperability by default; proper implementation is critical. Furthermore, as net-centric applications and services emerge, so too will the need to dynamically configure the IA Components to permit net-centric operations. As an example, **access control** based on **Internet Protocol** (IP) address would not work, as the addresses of service users will not be known a priori when such services are dynamically discoverable.

The DoD provides requirements and extensive guidance for the implementation of information assurance at the DISA Information Assurance Support Environment (IASE) Web site. In particular, the Network **Security Technical Implementation Guide** (STIG) on the IASE Web site provides guidance for the network implementation, particularly the boundary between the Node's internal network and external networks. It identifies several IA systems, capabilities, and configurations as listed below and provides guidance for implementation of each.

Rather than repeating the contents of specific guidance in this document, readers should check the IASE Web site for current Network IA guidance on topics such as the following:

- External Network **Intrusion Detection System** (IDS), anomaly detection, or prevention device if required by the **Computer Network Defense Service Provider** (CNDSP)
- **Router** Security with **Access Control Lists**
- **Firewall** and application level **proxies** (may be separate device to proxy applications)
- Internal **Network Intrusion Detection** (NID) system
- DMZ, if applicable for publicly accessible services
- Split Domain Name Service (DNS) architecture
- Domain Name System Security Extensions (DNSSEC) for higher level domain servers
- Secure devices and operating systems (i.e., **STIG** compliant)
- Ports and **protocols**

Furthermore, DoD **computer network defense** (CND) policies ***mandate all owners of DoD information systems and computer networks enter into a service relationship with a CNDS provider***.

Best Practices

- BP1701: Configure **Components** for **Information Assurance** (IA) in accordance with the Network **Security Technical Implementation Guide** (STIG).

P1138: Node Transport

A **Node** provides a transport infrastructure shared among the **components** within the Node, implements **Global Information Grid** (**GIG**) **Information Assurance** (**IA**) boundary protections, and is **Internet Protocol Version 6** (**IPv6**) capable. In some cases, guidance may seem rudimentary, but history demonstrates that configuration errors for such rudimentary aspects are often the cause of interoperability, integration, and IA issues.

Transport elements a Node provides are obviously essential in achieving net-centricity, but they also play a key role in minimizing interoperability issues.

Security Considerations

The **DISA Security Technical Implementation Guides** (**STIGs**; http://iase.disa.mil/stigs/stig/index.html) are applicable in several places throughout the NESI Part 4 Node Transport perspectives. The STIGs frequently change to include newly discovered vulnerabilities and as the current "state of the art" is refined. Consult the program-applicable STIGs and monitor them periodically for updates as a fundamental part of design activities.

For an overview of general security considerations, see the Enterprise Security [P1332] perspective. For additional detail, see the Data, Application and Service Integrity [P1338] perspective.

Management Considerations

For general management considerations, see the Security and Management [P1331] and Enterprise Management [P1330] perspectives. For additional detail, see the following perspectives:

- Design Tenet: Decentralized Operations and Management [P1276]
- Design Tenet: Enterprise Service Management [P1278]
- Design Tenet: Differentiated Management of Quality-of-Service [P1265]
- Traffic Management [P1356]

Detailed Perspectives

Transport elements that a Node provides are obviously essential in achieving net-centricity but also play a key role in minimizing interoperability issues. The following perspectives describe several Transport elements:

- Physical and Data Link Layers [P1348]
- Network layer [P1349]
- Transport Layer [P1350]
- Subnets and Overlay Networks [P1351]
- Network Services [P1353]
- Application Layer Protocols [P1355]
- Mobility [P1141]
- Traffic Management [P1356]

Guidance

- G1584: Provide a transport infrastructure that is shared among **components** within the Node.
- G1585: Provide a transport infrastructure for the Node that implements **Global Information Grid** (**GIG**) **Information Assurance** (**IA**) boundary protections.

Best Practices

- BP1704: Consult the applicable **Security Technical Implementation Guidance** (**STIG**) documents as a fundamental part of design activities, and monitor the STIGs periodically for updates.

P1348: Physical and Data Link Layers

As data flows to and from a computer (typically via Ethernet although there are other choices like asynchronous transfer mode or ATM; Sonet; and the IEEE 802.11 family) it moves through a modulator-demodulator device. This device structures the data into electronic signals that can be carried over physical communications media. This communication media may include copper wire, fiber optic cable, or wireless (such as microwaves, laser, or radio waves).

The data link layer is responsible for encoding bits into packets prior to transmission and then decoding the packets back into bits at the destination. Bits are the most basic unit of information in computing and communications. Packets are the fundamental unit of information transport in all modern computer networks, and increasingly in other communications networks as well.

The data link layer is also responsible for logical link control, media access control, hardware addressing, error detection and handling and defining physical layer standards. It provides reliable data transfer by transmitting packets with the necessary synchronization, error control and flow control.

The data link layer is divided into two sublayers: the media access control (MAC) layer and the logical link control (LLC) layer. The former controls how computers on the network gain access to the data and obtain permission to transmit it; the latter controls packet synchronization, flow control and error checking.

The data link layer is where most local area network (LAN) and wireless LAN technologies are defined. Popular technologies and protocols generally associated with this layer include the following.

- Ethernet
- Token Ring
- FDDI (fiber distributed data interface)
- ATM
- SLIP (serial line Internet protocol)
- PPP (point-to-point protocol)
- HDLC (high level data link control)
- ADCCP (advanced data communication control procedures).

Descriptions of a few of the possible standards and media follow.

IEEE 802 Standards

The services and protocols specified in IEEE 802 map to the lower two layers (Data Link and Physical) of the seven-layer Open Systems Interconnection (OSI) networking reference model. In fact, IEEE 802 splits the OSI Data Link Layer into two sub-layers named Logical Link Control (LLC) and Media Access Control:

- Data link layer
  - LLC Sublayer
  - MAC Sublayer
- Physical layer

Fiber Optic

Fiber optic related standards include the following.

- FDDI: ANSI X3T9.5 (Fiber Distributed Data Interface)
- SDH: ITU G.707 & G.708 SDH (Synchronous Digital Hierarchy; international form of SONET) SONET: Telcordia GR-253-CORE (Synchronous Optical Networking; Bell System form of SDH)
- ANSI T1.105-1991, *Digital Hierarchy - Optical Interface Rates and Formats Specification (SONET)*

- Fibre Channel: ANSI NCITS T11 (formerly X3T9.3) (mostly for storage area networks or SANs)
- GIG Ethernet: IEEE 802.3-2005 (also known as 802.3z; the fiber optic variants collectively are known as 10000BASE-X)

Tactical Data Links (TDL)

Joint Staff approved, standardized wireless/radio communications links suitable for transmission of digital information. Current practice is to characterize a tactical data link by its standardized message formats and transmission characteristics. TDLs interface two or more command and control or weapons systems via a single or multiple network architecture and multiple communications media for exchange of tactical information. Examples are Link 16 and **Situation Awareness Data Link** (**SADL**).

For more information see the Integration of Non-IP Transports [P1151] perspective.

SensorNets

A sensor network, or SensorNet is a network consisting of spatially distributed autonomous devices using sensors to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations cooperatively. More simply stated, it is a network where the source data is sensor data. SensorNets are often wireless networks. Wireless SensorNets can use any type of radio transmission on any protocol but most frequently employ **IP** data transfer.

Radio/Waveforms

IP network traffic can be conveyed over any radio. The legacy serial transmissions easily send and receive packets. Formatted radios such as Link-16 and others can also transfer packets but the packets must be "fit" into the format structure.

With the rise of software defined radios, the **NetOps** administrator or commander has the opportunity to select dynamically the kind of media communications technology most appropriate for use in the local sub-network infrastructure. This enables matching the Quality of Service (QoS) and **Information Assurance** goals to the underlying capabilities of the media communications.

A software defined radio (SDR) can receive or transmit signals in the radio frequency (RF) spectrum, but its signal-modulation methods depend on software loaded into the radio. Today, SDRs rely mainly on traditional circuits to process RF signals; but day by day, software gets closer to the antenna. A typical SDR comprises RF front-end circuits that connect to analog-to-digital converters (ADCs) on the receive side and digital-to-analog converters (DACs) on the transmit side. These converters connect to a signal processing subsystem that contains general-purpose or reconfigurable processors.

The processor software implements wireless standards, or "waveforms," such as Global System for Mobile communications (GSM), Code Division Multiple Access (CDMA) or the Single Channel Ground and Airborne Radio System (SINCGARS.) As long as the RF front-end circuits and the ADCs and DACs operate with a wide enough bandwidth, designers can modify the radio's capabilities simply by updating its software.

The **Joint Tactical Radio System** (**JTRS**) is a family of software-programmable tactical radios. They will provide combat personnel with voice, data, and video communications that are interoperable among all battlefield participants regardless of the branch of service.

In the case of a serial radio it will transfer packets at its designed channel data rates. So a 56,000 bits per second (56k bps) modem that is interfaced to a 56k bps radio or telephone line channel will transfer data at 56k bps. In the case of formatted radios this is not necessarily true. For example a user of a time slotted radio who has only one time slot every 12 seconds will have available the data rate in the time slot in bps divided by 12. Thus, these types of radios will change network performance.

P1349: Network Layer

The network layer is the third layer of seven in the Open Systems Interconnection (OSI) model [R1256] and the third layer of five in the **TCP/IP** model. These reference models are stacked architectures which allow separation of functions and thus make it easier from the software point of view to insert, replace, and separate software functional modules. In all of the models, the network layer responds to service requests from the transport layer and issues service requests to the data link layer.

In essence, the network layer is responsible for end-to-end (source-to-destination) packet delivery, whereas the data link layer is responsible for node-to-node (hop-to-hop) frame delivery.

The network layer provides the functional and procedural means of transferring variable length data sequences from a source to a destination via one or more networks while maintaining the quality of service and error control functions.

Detailed Perspectives

- Internet Protocol [P1139]
- IP Routing and Routers [P1143]
- Integration of Non-IP Transports [P1151]

P1139: Internet Protocol (IP)

The commercial **Internet** and U.S. Department of Defense (DoD) networks are built upon the **Internet Protocol** (**IP**). Today, these networks are based on version 4 of this protocol (IPv4). The primary motivation for embracing the next generation of IP (version 6 or IPv6) is due to the explosive growth of the Internet. The **Assistant Secretary of Defense for Networks and Information Integration**, **ASD(NII)**, has a goal which includes adapting Internet and **World Wide Web** constructs and standards with enhancements for mobility, surety, and military unique features (e.g., precedence, preemption) as one of nine *Net-Centric Attributes* [R1180]. IP is among the most fundamental of protocols needed for **Global Information Grid** (**GIG**) interoperability. There are, however, a number of interoperability challenges emerging as DoD usage of IP networking continues to expand.

IPv4

IPv4, the first widely deployed version of the Internet Protocol, currently is the dominant network layer protocol on the Internet and, apart from IPv6, it is the only standard internetwork-layer protocol used on the Internet. The Internet Engineering Task Force (IETF) described IPv4 in a September 1981 Request for Comments (IETF RFC 791). DoD also standardized IPv6 as MIL-STD-1777 dated 12 August 1983 (canceled 5 December 1995).

IPv4 is a data-oriented protocol for use on packet switched internetworks (e.g., Ethernet). It is a best effort protocol in that it does not guarantee delivery. IPv4 also does not make any guarantees on the correctness of the data; this may result in duplicated packets or packets delivered out of order. An upper layer protocol (e.g., **TCP** or, in part, **UDP**) needs to address these aspects.

Broadcast, Multicast

In computer networking, broadcasting refers to transmitting a packet that (conceptually) every device on the network will receive. In practice, the scope of the broadcast is limited to a broadcast domain. IPv4 supports broadcast, but IPv6 does not include it in the newer standard.

Multicast is the delivery of information to a group of destinations simultaneously using the most efficient strategy to deliver the messages over each link of the network only once, creating copies only when the links to the destinations split. As opposed to broadcast, multicast only sends information to a limited set of destinations.

IPv6

The Internet has been growing at an exponential rate, roughly doubling in size every year. Devices connected to the Internet are assigned globally unique addresses, and the available address space is rapidly becoming exhausted. IPv4 uses 32-bit addresses, constraining the number of unique addresses available as public Internet addresses; an IPv4 address shortage is inevitable. The IETF, to solve the address shortage problem and to provide other IP improvements, embarked on developing IPv6 to replace IPv4 after a long dual use transition period. IPv6 is already widely used in Asia, and manufacturers sell dual stack routers which process both IPv4 and IPv6 stacks.

IPv6 development supports the continued growth of the Internet by using 128-bit addresses to provide essentially unlimited address space. In addition, other improvements were made relative to IPv4, based on a generation of experience. Some of these other improvements are listed below:

- ***Streamlined processing within routers*** - The IPv6 protocol has a simplified header and the larger address allows summarizing routes in a hierarchical manner. This can dramatically reduce the size of routing tables and improve the performance of routers. IPv6 tries to make it easier to build very fast routers. IPv6 has no header checksum for routers to update, has no fragmentation in routers, has no options in the basic IPv6 header, and has a 64-bit word size.

- ***More efficient multicast support*** - All IPv6 implementations must support multicast. In addition, an added capability limits the scope of multicast transmissions. The addition of anycast addresses to IPv6 is a major development because anycast messages go only to one member of a defined group of multiple addresses, rather than to each member.

- ***Native mobility support*** - IPv6 has increased support for mobility and ad hoc networking, which is lacking or limited in IPv4. The IPv6 protocol provides an improved version of Mobile IP, which allows mobile computers to connect to the network at different locations without disrupting communications (elimination of "triangle routing" for mobile IP).

- ***Mandatory security features*** - All IPv6 implementations must support the IP Security (IPsec) features for data integrity and confidentiality (end-to-end, IP-layer authentication and encryption are possible). IPsec is available but optional for IPv4.

- ***Autoconfiguration*** - It is possible to configure the IP addresses and other network-related parameters automatically with or without separate servers. While IPv4 does have **Dynamic Host Configuration Protocol** (**DHCP**), some applications, such as IP Telephony, cannot operate through DHCP and DHCP is not scalable.

- ***Improved Neighbor Discovery*** - The IPv6 Neighbor Discovery (ND) provides a number of significant improvements over the IPv4 Address Resolution Protocol (ARP). ARP worked as a link-layer protocol using network broadcasts which link-layer bridges forward. For large subnets, ARP sometimes creates "broadcast storms" crowding out all useful network traffic for some period of time. Also ARP is insecure; there is no way to verify that a machine responding to an ARP query really is the correct machine; the result is that it is easy to steal traffic destined to another machine. ND on the other hand runs over IPv6 using multicasting, which is media independent. It is possible to constrain ND to where it is needed so as not to create broadcast storms. ND can work with IP Security to get authenticity and/or confidentiality guarantees.

- ***Hierarchical Addressing and Route Summarization*** - The IPv6 addressing structure differs significantly from IPv4. IPv6 supports improved hierarchical addressing with route summarization, address renumbering and multi-homed sites. These features have the potential to simplify network configurations and reconfigurations. Route summarization permits routers to exchange much less reachability information over the network, reducing router overhead traffic. This is of obvious benefit for tactical RF links. IPv4 already realizes some benefits of route summarization through a combination of Classless Interdomain Routing (CIDR) and hierarchical network assignments. IPv6 hierarchical addressing may require considerable adaption for mobile, multi-hop networks that involve movement across subnets. A more detailed analysis is needed to assess the value of hierarchical addressing in IPv6 for DoD mobile networks and RF subnets.

Additional IPv6 Information Sources

The following IETF Request For Comments documents represent a few of the RFCs available via the IETF RFC Index (created on 14 March 2009; http://www.ietf.org/iesg/1rfc_index.txt).

- RFC 4291, Draft Standard, *IP Version 6 Addressing Architecture*, February 2006
- RFC 3587, Informational, *IPv6 Global Unicast Address Format*, August 2003
- RFC 2375, Informational, *IPv6 Multicast Address Assignments*, July 1998
- RFC 2460, Draft Standard, *Internet Protocol, Version 6 (IPv6) Specification*, December 1998
- RFC 4861, Draft Standard, *Neighbor Discovery for IP version 6 (IPv6)*, September 2007
- RFC 4862, Draft Standard, *IPv6 Stateless Address Autoconfiguration*, September 2007
- RFC 4443, Draft Standard, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*, March 2006

Detailed Perspective

The following perspective addresses transition from IPv4 to IPv6:

- IPv4 to IPv6 Transition [P1140]

Guidance

- G1600: Obtain **Internet Protocol Version 6** (**IPv6**) addresses to use for DoD IP addressable resources from **DISA**.

P1140: IPv4 to IPv6 Transition

A 9 June 2003 **ASD(NII)**/DoD **CIO** memo, *Internet Protocol Version 6 (IPv6)*, [R1190] is the first in a series of memos addressing DoD transition to **IPv6** and establishing IPv6 as the next generation network protocol for DoD. The transition goal originally was Government FY 2008; however, transition planning is still under way. The DoD IPv6 Transition Office in the **Defense Information Systems Agency** (**DISA**) is responsible for master transition plan development, acquiring **Internet Protocol** (**IP**) addresses, providing necessary infrastructure and technical guidance, and ensuring the use of unified solutions across DoD to minimize cost and interoperability issues. DoD components are developing component transition plans and are providing guidance and governance to programs. There are Milestone Objectives (MOs) outlined for the gradual and controlled transition of the **enterprise**. Currently only those systems approved as MO1 pilots are allowed to switch to IPv6 in operational environments.

To enable this transition, as of 1 October 2003 all **Global Information Grid** (**GIG**) assets being developed, procured, or acquired shall be IPv6 capable (while retaining compatibility with IPv4). The ***DoD IPv6 Working Group*** is coordinating IPv6 implementation issues through formal standards bodies. A list of the standard IPv6 specifications approved for use in DoD networks so that they become "IPv6 capable" is in the **Defense IT Standards Registry** (**DISR**).

The working group tasks include preparing an IPv6 transition plan for the Node infrastructure as well as the transport users within the Node in coordination with the **Component** and DoD transition plan; the Node IPv6 transition plan is subject to review and approval by the appropriate IPv6 transition authority. Coordination is essential to ensure that the intermediate network infrastructures are IPv6 capable in the planned timeframe, and similarly for other-end network infrastructures for known system interfaces. The Node's IPv6 transition plan should consider applicable DoD Component IPv6 transition plans, IPv6 working group products, and interoperability testing. The net-centric concepts of loose coupling and discoverable services may be impacted by the transition to IPv6 if services begin depending on IPv6-specific features. Identify services which utilize IPv6 features and which may perform differently if accessed via an **Internet Protocol Version 4** (**IPv4**) infrastructure.

IPv6 transition has an impact on many transport infrastructure components. The IPv6 Transition Plan for a Node should include transition of all impacted network elements including the **Domain Name System** (**DNS**), routing, security, and dynamic address assignment.

The transition between today's IPv4 Internet and a future IPv6-based one will be a long process during which both protocol versions will coexist. The **Internet Engineering Task Force** (**IETF**) created the NGTrans Working Group (now concluded) to identify IPv6 transition issues and propose technical solutions to achieve it. Ongoing IPv6 operations standards, tools, techniques and best practices derived from both this work and experience with the 6bone testbed (also now retired) are the responsibility of the V6Ops Working Group.

No single general rule applies to the IPv4 to IPv6 transition process. In some cases, moving directly to IPv6 will be the answer. For instance IPv6 could be pushed by a political decision to extend the number of IP addresses to sustain the economic growth of a country. Another example is the large-scale deployment of a new IP architecture (such as mobile or home networking) to provide disruptive applications and innovative services.

Other transition plans will enable a gradual interoperability between IPv4 and IPv6 as transition evolves. Here, Internet Service Providers (ISPs) and enterprises will prefer to preserve the heavy investments made to deploy IPv4 networks.

Some studies foresee that the transition period will last between today and 2030-2040. At that time, IPv4 networks should have totally disappeared.

The NGTrans Working Group defined three main transition techniques.

- ***Dual-stack network***. The **dual stacking** approach requires hosts and routers to implement both IPv4 and IPv6 protocols. This enables networks to support both IPv4 and IPv6 services and applications during the transition period in which IPv6 services emerge and IPv6 applications become available. At the present time, the dual-stack approach is a fundamental mechanism for introducing IPv6 in existing IPv4 architectures and will remain heavily used in the near future. The drawback is that an IPv4 address must be available for every dual-stack machine. This is unfortunate, since IPv6 was developed precisely due to the scarcity of IPv4 addresses.

- ***Tunneling***. **Tunneling** enables the interconnection of IP clouds. For instance, a tunnel can interconnect separate IPv6 networks through a native IPv4 service. A border router encapsulates IPv6 packets before transportation across an

IPv4 network and decapsulates the packets at the border of the receiving IPv6 network. Tunnel configuration can be static, dynamic, or implicit (6to4, 6over4). The Tunnel Broker (TB) approach automatically can manage tunnel requests coming from the users and ease the configuration process. The Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) is a recent technique to avoid manual tunnel configuration. In later stages of transition, tunnels will also interconnect remaining IPv4 clouds through the IPv6 infrastructure.

- ***Translation mechanism***. Translation is necessary when an IPv6 only host has to communicate with an IPv4 host. At the least, the IP header requires translation, but the translation will be more complex if the application processes IP addresses; in fact such translation inherits most of the problems of IPv4 network address translators. Application-Level Gateways (ALGs) translate embedded IP addresses, recompute checksums, etc. Stateless IP/ICMP Translation (SIIT) and Network Address Translation-Protocol Translation (NAT-PT) are the associated translation techniques. A blend of translation and the dual stack model, known as Dual Stack Transition Mechanism (DSTM), addresses the case where insufficient IPv4 addresses are available. Like tunneling techniques, translation implementation can be in border routers and hosts.

There are many ways to "mix and match" this complex set of coexistence and transition techniques.

Guidance

- G1586: Provide a transport infrastructure for the Node that is **Internet Protocol Version 6** (**IPv6**) capable in accordance with the appropriate governing transition plan.
- G1587: Prepare an **Internet Protocol Version 6** (**IPv6**) transition plan for the Node.
- G1588: Coordinate an **Internet Protocol Version 6** (**IPv6**) transition plan for a Node with the **Components** that comprise the Node.
- G1589: Address issues in the appropriate governing **Internet Protocol Version 6** (**IPv6**) transition plan as part of the IPv6 Transition Plan for a Node.
- G1590: Include transition of all the impacted elements of the network as part of the **Internet Protocol Version 6** (**IPv6**) Transition Plan for a Node.
- G1591: Prepare IPv6 Working Group products as part of the **Internet Protocol Version 6** (**IPv6**) transition plan for a Node.
- G1592: Include interoperability testing in the plan as part of the **Internet Protocol Version 6** (**IPv6**) transition plan for a Node.
- G1599: Simultaneously support **Internet Protocol Version 4** (**IPv4**) and **Internet Protocol Version 6** (**IPv6**) in the Node's **Domain Name System** (**DNS**) service.
- G1600: Obtain **Internet Protocol Version 6** (**IPv6**) addresses to use for DoD IP addressable resources from **DISA**.

Best Practices

- BP1705: Design **Domain Name System** (**DNS**) infrastructure in accordance with appropriate governing **Internet Protocol Version 6** (**IPv6**) Transition Office requirements.
- BP1923: Employ an operating system that supports simultaneously IPv4 and IPv6.

P1143: IP Routing and Routers

Routers not only provide the main connection to the **Global Information Grid** (**GIG**), but they also are a first line of **computer network defense**. These complex devices provide security filtering, address management, network management, and time synchronization. A ***GIG Router Working Group (GRWG)*** is addressing implementation issues.

Components should be able to operate in a heterogeneous environment. The presence of **Internet Protocol Version 4** (**IPv4**) and **Internet Protocol Version 6** (**IPv6**) packets and services in a dual-stack environment should not cause a degradation of application performance.

Routing capabilities in real-time, dynamic and mobile environments, such as at the tactical edge, are still in their infancy. A variety of working groups, such as the GRWG and the Office of the Secretary of Defense ***Joint Airborne Network (JAN) Working Group***, continue to define, prototype and refine routing capabilities.

Routing is an umbrella term for the set of protocols that determine the path that data follows in order to travel across multiple networks from a source to a destination. Data routing from source to destination is through a series of routers and across one or more networks.

Routing protocols enable a router to build up a forwarding table that correlates final destinations with next hop addresses. Routing protocols specify a set of messages routers exchange; the message contents allow a router to inform its peers about the **IP** routes it knows and allow that knowledge to spread throughout the network.

An IP network administered by a single authority is called an autonomous system (AS); such a network could run an Interior Gateway Protocol (IGP). However, multiple autonomous systems also need to interconnect and exchange routes among themselves to create a larger network not administered by any single authority; the public **Internet** is an example. In this case selecting routes to add to the IP forwarding table requires great flexibility; for example, path length may not be meaningful if part of that path has links with costs set by a different AS using different criteria. More important are administrative policies like the selection of preferred transit networks with which to partner. The Border Gateway Protocol (BGP) serves this environment. It allows each AS to select which other AS are the preferred choices to inject routes into its network.

When BGP routers propagate an IP route to another AS, they include the entire list of AS that have propagated the route to them, from the AS that originated the route to the current AS propagating it further. This is called the path vector and BGP is a path vector protocol. Having the entire list of AS that have propagated the route allows a BGP router to decide if the route uses its preferred transit AS or goes through an AS to avoid whenever possible. This is greater flexibility than offered by a shortest path IGP. Note that IP networking requires loop-free paths but not necessarily shortest paths; the BGP path vector guarantees loop-free paths.

Example routing protocols follow.

Open Shortest Path First (OSPF) Protocol

The OSPF protocol is a hierarchical interior gateway protocol (IGP) for routing in Internet Protocol, using a link-state in the individual areas that make up the hierarchy. The protocol uses a computation based on Dijkstra's algorithm to calculate the shortest path tree inside each area. OSPF is the primary means of routing in the Internet. It does not respond well to rapidly changing node connectivity and as such is not considered to be suitable for mobile, wireless military networks.

The following **Internet Engineering Task Force** (**IETF**) Requests For Comments (RFCs) provide additional information concerning OSPF:

- RFC 2328, Standard, *OSPF Version 2*, April 1998, for unicast routing
- RFC 3101, Proposed Standard, *OSPF Not-So-Stubby Area (NSSA) Option*, January 2003
- RFC 1793, Proposed Standard Extending OSPF to Support Demand Circuits, April 1995; updated by RFC 3883, *Proposed Standard, Detecting Inactive Neighbors over OSPF Demand Circuits (DC)*, October 2004
- RFC 5340, Proposed Standard, *OSPF for IPv6*, July 2008
- RFC 3137, Informational, *OSPF Stub Router Advertisement*, June 2001

- RFC 3630, Proposed Standard, *Traffic Engineering (TE) Extensions to OSPF Version 2*, September 2003; updated by RFC4203, Proposed Standard, *OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)*, October 2005
- RFC 1584, Historic, *Multicast ExtensionstoOSPF*, March 1994
- RFC 1585, Informational, *MOSPF: Analysis and Experience*, March 1994

Border Gateway Protocol (BGP)

BGP is the standard protocol for routing between autonomous system (AS) domains. It works by maintaining a table of IP networks or "prefixes" which designate network reachability among autonomous systems. It relies on **Transmission Control Protocol** (**TCP**) sessions between BGP peers and does not have an automatic neighbor discovery capability. As the number of AS domains increases, BGP may take longer to converge than OSPF after a routing change occurs.

The following IETF RFCs provide additional BGP information:

- RFC 4271, Draft Standard, *Border Gateway Protocol 4 (BGP-4)*, January 2006
- RFC 1772, Draft Standard, *Application of Border Gateway Protocol In the Internet*, March 1995
- RFC 4760, Draft Standard, *Multiprotocol Extensions for BGP-4*, January 2007
- RFC 3107, Proposed Standard, *Carrying Label Information in BGP-4*, May 2001
- RFC 5065, Draft Standard, *Autonomous System Configurations for BGP*, August 2007
- RFC 2439, Proposed Standard, *BGP Route Flap Damping*, November 1998
- RFC 4659, Proposed Standard, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*, September 2006
- RFC 4797, Informational, *Use of Provider Edge to Provider Edge (PE-PE) Generic Routing Encapsulation (GRE) or IP in BGP/MPLS IP Virtual Private Networks*, Jan 2007
- RFC 4456, Draft Standard, *BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*, April 2006
- RFC 4384, Best Current Practice, *BGP Communities for Data Collection*, February 2006

Routing Information Protocol (RIP)

RIP sends routing-update messages at regular intervals and when the network topology changes. When a router receives a routing update that includes changes to an entry, it updates its routing table to reflect the new route. The metric value for the path increases by 1, and the sender is the next hop. RIP routers maintain only the best route (the route with the lowest metric value) to a destination. After updating its routing table, the router immediately begins transmitting routing updates to inform other network routers of the change. These updates are sent independently of the regularly scheduled updates that RIP routers send.

Intermediate System - Intermediate System Protocol

The IS-IS protocol is one of a family of IP routing protocols. IS-IS is an Interior Gateway Protocol (IGP) for the Internet, used to distribute IP routing information throughout a single Autonomous System (AS) in an IP network.

IS-IS is a link-state routing protocol, which means that the routers exchange topology information with their nearest neighbors. The topology information is flooded throughout the AS, so that every router within the AS has a complete picture of the topology of the AS. This picture is then used to calculate end-to-end paths through the AS, normally using a variant of the Dijkstra algorithm. Therefore, in a link-state routing protocol, the next hop address to which data is forwarded is determined by choosing the best end-to-end path to the eventual destination.

Additional information sources include the following:

- IETF RFC 1142, Informational, *OSI IS-IS Intra-domain Routing Protocol*, February 1990
- IS-IS Protocol: Intermediate System - Intermediate System, http://www.dataconnection.com/iprouting/isisprotocol.htm

Internet Control Message Protocol (ICMP)

ICMP is a network layer Internet protocol that provides message packets to report errors and other information regarding IP packet processing back to the source. IETF has documented ICMP in RFC 792, *Internet Control Message Protocol*, September 1981.

ICMP generates several kinds of useful messages, including Destination Unreachable, Echo Request and Reply, Redirect, Time Exceeded, Router Advertisement, and Router Solicitation. If an ICMP message cannot be delivered, the message is not retransmitted to avoid an endless flood of ICMP messages.

ICMP Router-Discovery Protocol (IDRP)

IDRP uses Router Advertisement and Router Solicitation messages to discover the addresses of routers on directly attached subnets. Each router periodically multicasts Router Advertisement messages from each of its interfaces. Hosts then discover addresses of routers on directly attached subnets by listening for these messages. Hosts can use Router-Solicitation messages to request immediate advertisements rather than waiting for unsolicited messages.

IRDP offers several advantages over other methods of discovering addresses of neighboring routers. Primarily, it does not require hosts to recognize routing protocols, nor does it require manual configuration by an administrator.

Guidance

- G1601: Use configurable **routers** to provide dynamic **Internet Protocol** (**IP**) address management using the **Dynamic Host Configuration Protocol** (**DHCP**).
- G1602: Use configurable **routers** to provide static **Internet Protocol** (**IP**) addresses.
- G1604: Use configurable **routers** to provide time synchronization services using **Network Time Protocol** (**NTP**).
- G1605: Use configurable **routers** to provide **multicast** addressing.
- G1606: Manage **routers** remotely from within the **Node**.
- G1607: Configure routers according to **National Security Agency** (**NSA**) Router Security Configuration guidance.

Best Practices

- BP1699: Configure **routers** in accordance with the Network **Security Technical Implementation Guide** (**STIG**).
- BP1700: Configure **routers** in accordance with Enclave **Security Technical Implementation Guide** (**STIG**).

P1151: Integration of Non-IP Transports

Systems that are not **Internet Protocol** (IP) networked, such as aircraft data links (**Link-16**, **SADL**, etc.), should implement IP gateways to interoperate with the **Global Information Grid** (GIG) until IP is supported natively. Most such systems already have plans for transition to IP networking, and gateways are an interim measure.

Implement these gateways as **services** in accordance with ***NESI Part 5: Developer Guidance***. This does not mean that the service would be limited to request/reply or other such usage patterns. In fact, for high-frequency data, such as track reporting, a function of the service could be to set up an out-of-band communication with a subscriber.

Guidance

- G1611: Implement Internet Protocol (**IP**) gateways to interoperate with the **Global Information Grid** (**GIG**) until IP is supported natively for **Components** that are not IP networked.

P1350: Transport Layer

The Transport Layer traditionally is the fourth layer of the Open Systems Interconnection (OSI) Reference Model. It provides transparent transfer of data between end systems using the services of the network layer (e.g., **Internet Protocol** or **IP**) below to move packets of data between the two communicating systems.

Transmission Control Protocol (TCP)

TCP, one of the core protocols of the IP suite, provides guaranteed delivery of messages when required. TCP divides messages into packets which are acknowledged back to the sending computer. If a packet is not acknowledged TCP retransmits the package. There are many current variants of TCP; the most common is called TCP Reno. Others like TCP Westwood, TCP Peach, TCP Vegas, TCP Real, etc., address issues that TCP has with network congestion. Using TCP, programs on networked computers can create connections to one another, over which they can send data. The protocol guarantees that data the source sends will be received in the same order without any missing packets.

In addition to variants of TCP, extensions to TCP exist to optimize performance in networks with issues such as packet loss and high latency. These issues cause poor network performance when using TCP (due to issues with the TCP cumulative acknowledgment algorithm in this environment). One such extension is TCP Selective Acknowledgment (TCP SACK). TCP SACK is useful for networks where high packet loss is probable (or when packets arrive out of order), such as with mobile networks. TCP SACK attempts to increase network throughput by following a process of selective acknowledgment where the data receiver informs the sender about all segments that have arrived successfully. Thus, the sender may retransmit only the undelivered segments.

For further discussion of mobility considerations see the Mobility [P1141] perspective.

User Datagram Protocol (UDP)

UDP is a connectionless transport layer protocol that belongs to the Internet Protocol family. UDP is basically an interface between IP and upper-layer processes. Unlike TCP, UDP adds no reliability, flow-control, or error-recovery functions. However, UDP consumes less network overhead than TCP.

UDP is useful in situations where the reliability mechanisms of TCP are not necessary, such as in cases where a higher-layer protocol might provide error and flow control.

Space Communications Protocol Specifications (SCPS)

The Space Communications Protocol Specifications (SCPS) are a collection of communications protocols the Consultative Committee on Space Data Systems (CCSDS) developed to provide reliable communications in space environments. SCPS include file transfer, transport, security, and network protocols. For more information on these recommended standards, see the CCSDS Blue Books: Recommended Standards Web page.

- *Space Communications Protocol Specification (SCPS)-File Protocol (SCPS-FP)*, CCSDS 717.0-B-1, May 1999 [under consideration for removal from the CCSDS library due to lack of use at present]; ISO 15894
- *Space Communications Protocol Specification (SCPS)-Transport Protocol (SCPS-TP)*, CCSDS 714.0-B-2, October 2006; ISO 15893
- *Space Communications Protocol Specification (SCPS)-Security Protocol (SCPS-SP)*, CCSDS713.5-B-1, May 1999; ISO 15892
- *Space Communications Protocol Specification (SCPS)-Network Protocol (SCPS-NP)*, CCSDS 713.0-B-1, May 1999; ISO 1589

SCPS protocol suite development supports space channels where the round trip delay is high and the error rate can be higher than that seen on the wires and fibers used in ground networks employing **TCP/IP**. TCP has great difficulty with high error rates and high round trip delays. As a result, attempts to use alternatives including SCPS-TP commonly occur. However, using a substitute protocol creates accountability issues as it must tell the source that a message was delivered when it was not and it then takes responsibility for delivery. If ultimate delivery fails, the source does not get a final delivery notification; it gets a failure message and the sender must take an alternate action that is unexpected. Imagine tracking a time critical target, sending orders, and later finding out the orders were not delivered. For further information about the SCPS protocol suite see http://www.scps.org/.

P1351: Subnets and Overlay Networks

Subnets and overlay networks are both building blocks by which net-centric applications, data and **services** bind transport network resources to their particular needs.

The sections below cover some of the standard transport binding address-constructs, binding techniques and operational rationales used by applications, data, and services when binding to the transport infrastructure.

Subnets

Subnets are the original technique by which Internet host systems were grouped "close" together for performance and "within" security perimeters. Nodes on a subnet often also use a single media technology optimized for their local area, a **Local Area Network** (**LAN**).

Subnets are a way of structuring the network by grouping all systems that share a single local area media such as a broadband LAN, a wireless data link or fiber bundle that share a single subnet mask (**IPv4**) or prefix (**IPv6**).

A designated router represents each subnet in the larger **Global Information Grid** (**GIG**). This router is responsible for both tracking changes in the immediate global network topology and ensuring that local changes do not concern the larger GIG unless absolutely necessary.

Media Access Control (MAC) addressing and designated routers both can change as systems start up, move and shutdown; a key to successful network performance is ensuring that both addressing and router election are correct and efficient.

Subnet membership helps to ensure both information distribution performance and protection; sometimes there is a desire to extend the use of subnets beyond the normal range of a particular media. This can be accomplished through use of link layer device such as repeater or bridge, which like routers forward traffic but unlike routers do not concern themselves with the topology of the larger GIG or **IP** addresses.

Link layer devices may also serve as sub-sub-nets known as virtual local area networks or VLANs when, instead of extending the range of the local media, they partition a single local media such as broadband for performance or protection purposes. Subnets are also important for larger GIG resiliency because they enable multi-homing in which a local area network connects to the larger GIG through more than one subnet address space, represented by more than one designated router. These alternate connections create a mesh of alternate paths for traffic to use, enabling both failover capability and load-sharing.

Overlay Networks

Overlay networks are a virtual extension of the subnet concept, but instead of blocks of IP addresses they use other network identifier constructs. Formally, an overlay network is a virtual network built on top of another network. Nodes in the overlay are connected by virtual or logical links, each of which may run on top of many lower layer links in the underlying networks. Overlay networks can be created at any layer in the Transport stack, but their network location identifiers usually bind to an IP address. SPINES (see http://www.spines.org/) is an example open source general purpose overlay that can be readily tailored for various applications from the Distributed Systems and Networks lab at Johns Hopkins University.

Virtual Private Network (VPN) Overlay Networks

- ***MPLS VPNs*** - MultiProtocol Label Switching (MPLS) VPNs use special short-hand labels to create overlay networks that conform to more sophisticated forwarding policies than the default IP routing metrics. They are especially useful in limiting the variability of delay or choice of intermediate networks.
- ***IPSec VPNs*** - Internet Protocol Security (IPSec) VPNs use cryptography to tunnel sensitive information exchanges through less-trusted intermediate networks.

For further VPN content, see the Virtual Private Networks [P1149] perspective.

Content Delivery Overlay Networks

Content Delivery Overlay Networks are used for replication and synchronization; a content delivery network (CDN) is a multicast-address network that extremely efficiently distributes web content, especially for load-sharing or content with high QoS requirements such streaming audio, video, and Internet television (IPTV) programming. CDNs are, in the strictest sense, Network Layer Overlay Network because they are based on multicast addressing that is maintained by multicast-capable routers.

Application Layer Overlay Networks

The following techniques are example application layer overlay networks.

- ***P2P Overlays*** - Peer-to-peer networks are typically used for connecting nodes via largely ad hoc connections set up and labeled for each information flow of interest. These are used to build a distribution topology based on application layer protocols that advertise local availability of content. For further information on P2P concepts see http://en.wikipedia.org/wiki/Peer-to-peer.
- ***Content Routers*** - Message Router overlays match content (often represented as XML) needs to content suppliers, often through deep packet inspection that then generate the information flow labels, which are then used to select appropriate Network layer routes. In some implementations, content router(s) can distribute the content needs of all subscribers (e.g. applications and users) across the network and can optimally push the matching content to each subscriber upon publication.
- ***Disruption Tolerant Networking*** - DTN overlays use proxies to stand in for content suppliers and consumers whose network layer connectivity may be intermittent or changing. Information flow labels are assigned to either the current "best" network layer route or a temporary buffering server if one is not available. For an example of an application of DTN, see the *Disruption Tolerant Networking for Marine Corps CONDOR* paper from the Military Communications Conference, 2005 (MILCOM 2005).

Detailed Perspectives

- Broadcast, Multicast and Anycast [P1146]
- Virtual Private Networks [P1149]
- Ad Hoc Networks [P1352]

P1146: Broadcast, Multicast, and Anycast

Broadcast, **Multicast**, and Anycast are bandwidth optimizations techniques for content dissemination; they are all used to send packets of information from a source simultaneously to multiple destinations unlike Unicast which routes information from a source to a singe destination.

Broadcast

Broadcast delivers data to all addresses on a media; for example the various wired (802.3/Ethernet) and wireless (802.11/WiFi) broadcast mechanisms that use special addresses on which all host systems must receive messages. Broadcast implemetation may be at the link layer or at the network layer (available in **Internet Protocol Version 4**, or **IPv4**, but not **IPv6**) or higher layers.

Multicast

IP Multicast is the delivery of information to a group of destinations simultaneously using the most efficient strategy to deliver the messages over each link of the network only once, creating copies only when the links to the destinations split. Multicast currently supports various groups throughout the DoD to provide capabilities such as collaboration and alerting; the use of multicast addressing is growing. Multicast capability is being engineered actively into the **Global Information Grid** (**GIG**). Careful planning is still required, however, until multicast becomes ubiquitous across the entire GIG.

Anycast

Anycast (included as part of the formal IPv6 specification but implemented as external extensions to the IPv4 specification) is a network addressing and routing scheme to route data to the next router or next group of routers in a network. A combination of Anycast and Multicast can create the functionality of Broadcast in an IPv6 network.

Guidance

- G1601: Use configurable **routers** to provide dynamic **Internet Protocol** (**IP**) address management using the **Dynamic Host Configuration Protocol** (**DHCP**).
- G1610: Configure the **Dynamic Host Configuration Protocol** (**DHCP**) services to assign **multicast** addresses.

Best Practices

- BP1706: Design node networks, including the selection of **Components** and configuration, to support **multicasting** even if not currently used.

P1149: Virtual Private Networks (VPN)

Virtual Private Networks (**VPNs**) create a private "**tunnel**" within a network by encrypting traffic between specified end points. If a **Node** requires a VPN, implement it in accordance with the guidance provided in the Network **Security Technical Implementation Guide** (**STIG**). Do not place services and information intended to be broadly accessible to other **Global Information Grid** (**GIG**) Nodes behind a VPN because they will be reachable by only the Nodes that are part of the VPN.

A VPN is a private network overlaid on top of a public network (usually the **Internet**) to connect remote sites or users together. Instead of using a dedicated, real-world connection such as a leased line, a VPN uses "virtual" connections routed through the Internet from a private network (such as a company's intranet) to an authorized remote site or user (such as a company's employee that does not otherwise have direct access to the company's intranet).

The VPN overlay approach extends the subnetwork concept of using address assignment to run logical links over local media networks. Overlay VPN logical links run on top of any kind of network: local media, IP network or another overlay network. Such overlay nets and VPNs are usually optimized for performance or protection or both.

VPNs sometime use standards such as **High Assurance Internet Protocol Encryption** (**HAIPE**) and Internet Protocol Security (IPsec) for security.

Guidance

- G1667: Implement **Virtual Private Networks** (**VPNs**) in accordance with the guidance provided in the Network **Security Technical Implementation Guide** (**STIG**).

Best Practices

- BP1702: Do not place services and information intended to be broadly accessible to other nodes behind a **Virtual Private Network** (**VPN**).

P1352: Ad Hoc Networks

A wireless ad hoc network is a decentralized wireless network containing two or more participants. In some ad hoc networks, participants are willing to forward data for other participants, as in the case of Internet Connection Sharing or Mobile Ad Hoc Network (MANET). Sometimes ad hoc networks (including MANET), determine dynamically which participants forward data based on the network connectivity. This is in contrast to wired networks, in which routers perform the task of routing, and managed wireless networks, in which a special node known as an access point manages communication among other nodes.

Commercial routing protocols, such as Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP), are designed and optimized for fixed infrastructures. The frequency of the message intervals to locate neighbor nodes and exchange routing tables is too low to keep up with the dynamic and mobile network state in a mobile environment or other similar unstable environments. An **Internet Protocol** (**IP**) routing protocol for mobile environments needs to interoperate with standard routing technology, detect and adapt to recurring link failures and mobility with minimal overhead and route data over the platform's multiple links to maximize throughput and reliability. For each of these requirements, the academic and research communities have done related work in the areas of MANET, multipath routing, and wireless extensions to common routing protocols. Continued research is needed to determine the best protocol settings to use (link metrics, hello intervals, dead intervals, etc.) and how to modify/extend the standard protocols to meet the requirements for mobile environments.

A MANET is a wireless ad hoc network of mobile routers (and associated hosts) connected by wireless links, the union of which form an arbitrary topology. The routers are free to move randomly and organize themselves arbitrarily; thus, the network's wireless topology may change rapidly and unpredictably.

Individual mobile networks implement their own internal MANET routing protocols which are transparent to IP (i.e., Open Systems Interconnection [OSI] Layer 3) and do not extend across mobile network boundaries. However, these mobile networks can interface with other networks using standard routing protocols, such as the OSPF protocol and BGP.

Additional Information

The following book and Internet Engineering Task Force (IETF) Requests for Comments (RFCs) provide additional information:

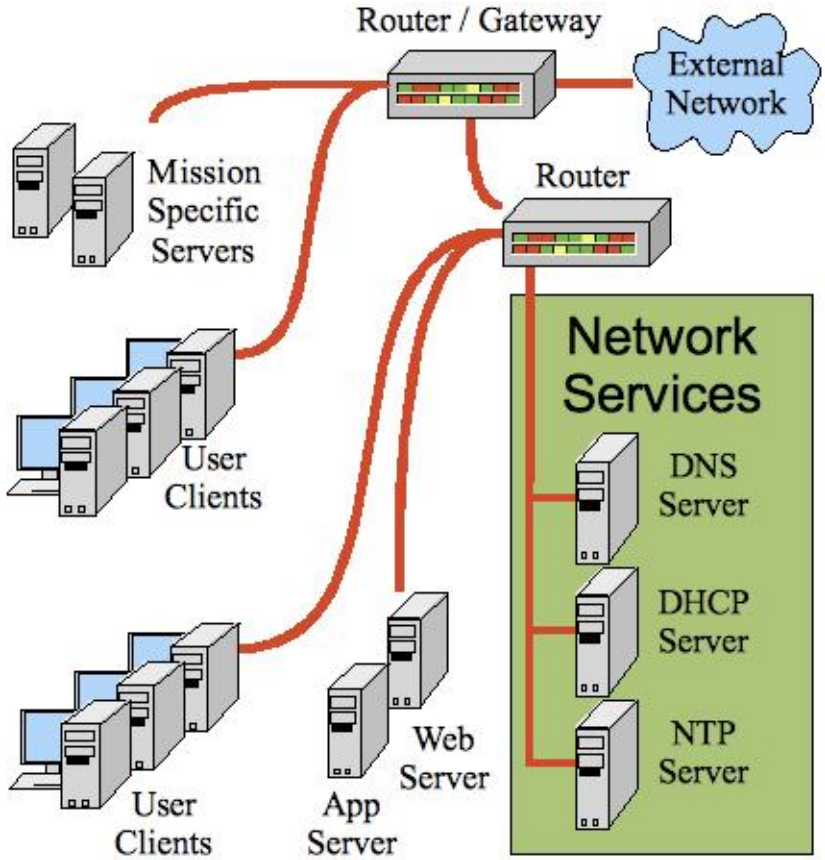
- C K Toh, *Ad Hoc Mobile Wireless Networks*, Prentice Hall Publishers, 2002.
- IETF RFC 3561, *Experimental Ad Hoc On Demand Distance Vector (AODV)*, July 2003
- IETF RFC 3684, *Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)*, February 2004
- IETF RFC 4728, *Experimental Dynamic Source Routing (DSR)*, Feb 2007
- IETF RFC 3626, *Experimental Optimized Link State Routing (OLSR)*, Oct 2003

P1353: Network Services

Network services are a special category of **services** available over **Internet Protocol** (**IP**) networks to network clients (hosts) that network administrators generally manage and maintain. When network clients request to join a network, they receive various configuration parameters that enable and facilitate the use of the network. The configuration parameter distribution can be manual (i.e., via paper) or via automated protocols. Regardless of the distribution mechanism, the network client must be configured accordingly.

Network service servers predominately provide services that are generic and local in nature. For example, the local network generally provides the time service. Some newer network services have replaced older versions (i.e., **Network Time Protocol** (**NTP**) time services have replaced Time Server services, and **Domain Name System** [**DNS**] has replaced the Name Server). Any service could theoretically be categorized as a network service; however, network services generally provide a service that is important for the integrity or security of the network and the safety of its clients.

Most network services are simply represented by the name of the service and an IP address. One major exception is the **Dynamic Host Configuration Protocol** (**DHCP**) server which is responsible for providing automated distribution of the configuration parameters. Access to this server is via a special broadcast message (DHCPDISCOVER) requesting membership onto the network. Most DHCP Clients know how to obtain from the DHCP Server the list of IP addresses that provide time using the DHCP options numbers.



I1220: Common Network Services

The following table list some of the more common configuration parameters that DHCP services provide as defined by the Internet Engineering Task Force Network Working Group in RFC 2132, *DHCPOptions and BOOTP Vendor Extensions*:

Configuration Parameter	Description

DNS Servers	The DNS option specifies a list of Domain Name System name servers available to the client; list servers in order of preference
NTP Servers	The NTP option specifies a list of IP addresses indicating NTP servers available to the client. Servers should be listed in order of preference
Trivial File Transport Protocol (TFTP) Server	The TFTP option identifies a TFTP server when using the "sname" field for DHCP options in the DHCP header

Detailed Perspectives

- Doman Name System [P1142]
- Dynamic Host Configuration Protocol [P1354]
- Network Time Service [P1144]

P1142: Domain Name System (DNS)

The **Domain Name System** (**DNS**) stores the relationships of host **Internet Protocol** (**IP**) address and their corresponding domain names in the equivalent of a distributed database (used here as a simplistic concept). The most import role of the DNS is to map IP addresses to human friendly domain names and back again. For example, where **`nesi.spawar.navy.mil`** may map to an **Internet Protocol Version 4** (**IPv4**) address of **`128.49.49.225`**, the **Internet Protocol Version 6** (**IPv6**) address might be **`1080::34:0:417A`**. For more information on DNS see the Internet Engineering Task Force (IETF) *Domain Names - Concepts and Facilities* Standard (RFC 1034). DNS also performs other essential functions, such as reverse lookups (obtaining host names from IP addresses, which can be important for security) and email configuration (special DNS **Mail eXchange (MX) Records** indicate the **server** used to receive email for a host). These capabilities are fundamental to net-centric operations and are essential for other computing, network, and **Enterprise Services**.

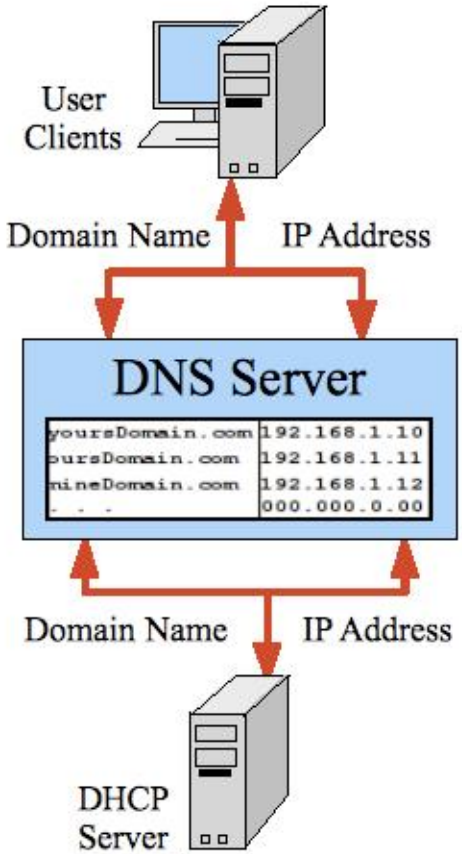
The DNS namespace is hierarchical. At each level in the hierarchy, the namespace can be divided into sub-namespaces called zones, which are delegated to other authoritative servers and which can be divided and delegated to other authoritative servers, and so on.

Each Node should implement DNS to manage hostname/address resolution within the Node, rather than use hard coded IP addresses, and use the DNS Mail eXchange (MX) Record capabilities to configure electronic mail delivery to the Node.

The DNS implementation should reflect the guidance provided in the *Domain Name System Security Technical Implementation Guide*. This **STIG** addresses implementation options such as the choice of basic DNS server types (primary, secondary, caching-only), use of a split-DNS design, location of servers in the network and relationship to other network entities, secure administration, security of zone transfers, and initial configuration.

Consider operational performance constraints, such as narrow bandwidth and intermittent connectivity, in designing the DNS for a **Node**. It may be desirable, for instance, to implement a caching-only DNS server for constrained environments.

The following image (I1221) shows a client requesting a domain name resolution as well as a **Dynamic Host Configuration Protocol** (**DHCP**) server updating DNS records.



I1221: DNS

Guidance

- G1662: Follow the guidance provided in the **Security Technical Implementation Guide** (**STIG**) for **Domain Name System** (**DNS**) implementations.
- G1595: Implement **Domain Name System** (**DNS**) to manage hostname/address resolution within the Node.
- G1596: Use **Domain Name System** (**DNS**) **Mail eXchange (MX) Record** capabilities to configure electronic mail delivery to the Node.
- G1598: Allow dynamic **Domain Name System** (**DNS**) updates to the Node's internal DNS service by local **Dynamic Host Configuration Protocol** (**DHCP**) **server(s)**.
- G1599: Simultaneously support **Internet Protocol Version 4** (**IPv4**) and **Internet Protocol Version 6** (**IPv6**) in the Node's **Domain Name System** (**DNS**) service.
- G1600: Obtain **Internet Protocol Version 6** (**IPv6**) addresses to use for DoD IP addressable resources from **DISA**.

Best Practices

- BP1597: Consider operational performance constraints in the design of the Node's **Domain Name System** (**DNS**).
- BP1663: Design a **Domain Name System** (**DNS**) in coordination with the appropriate governing **Internet Protocol Version 6** (**IPv6**) Transformation Office.
- BP1705: Design **Domain Name System** (**DNS**) infrastructure in accordance with appropriate governing **Internet Protocol Version 6** (**IPv6**) Transition Office requirements.

P1354: Dynamic Host Configuration Protocol (DHCP)

The **Dynamic Host Configuration Protocol** (**DHCP**) automates the network configuration of network devices (i.e., hosts) connected to **Internet Protocol** (**IP**) based networks. DHCP is built on the client-server model. A DHCP server allocates and manages IP addresses and delivers IP network configuration parameters (such as the default gateway, DNS servers, and other servers including time) to DHCP clients. DHCP consists of two major components:

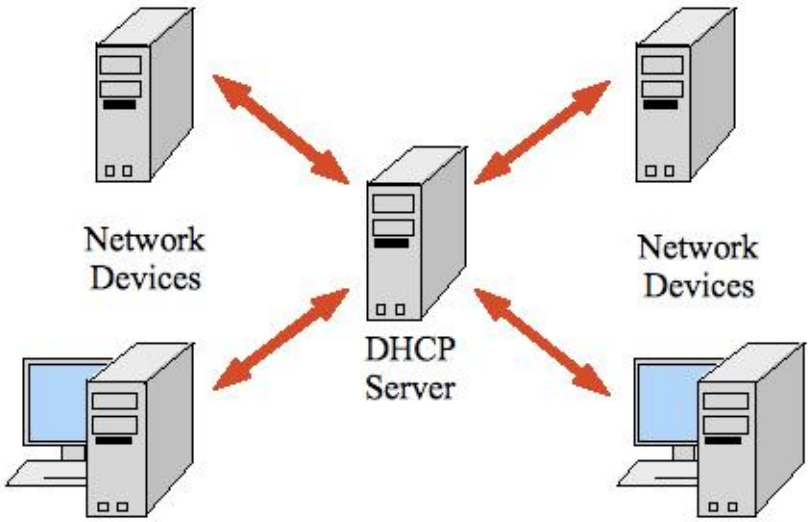
- A protocol for requesting and delivering to a DHCP client specific configuration parameters from a DHCP server
- A mechanism for managing and allocating IP addresses to DHCP clients

DHCP clients discover DHCP servers using a broadcast message rather than finding the DHCP servers in a directory. If there are multiple DHCP servers that hear the broadcast, they each can make an offer to the DHCP client to provide DHCP services. The client then chooses one of the offers; this provides a starting point for discovering all the other network services on the network.

DHCP provides three modes for allocating IP addresses. The best-known mode is ***dynamic***, in which the client receives a "lease" on an IP address for a period of time. Depending on the stability of the network, this could range from hours (a wireless network at an airport) to months (for desktops in a wired lab). At any time before the lease expires, the DHCP client can request renewal of the lease on the current IP address. A properly-functioning client will use the renewal mechanism to maintain the same IP address throughout its connection to a single network; otherwise, it may risk losing its lease while still connected, thus disrupting network connectivity while it renegotiates with the server for its original or a new IP address.

The two other modes for allocation of IP addresses are ***automatic*** (also known as DHCP Reservation), in which the address is permanently assigned to a client, and ***manual***, in which the address is selected by the client (manually by the user or any other means) and the DHCP protocol messages are used to inform the server that the address has been allocated.

Use of the automatic and manual methods generally is in situations which require finer-grained control over IP address (typical of tight firewall setups, although typically a firewall will allow access to the range of IP addresses that the DHCP server can allocate dynamically).

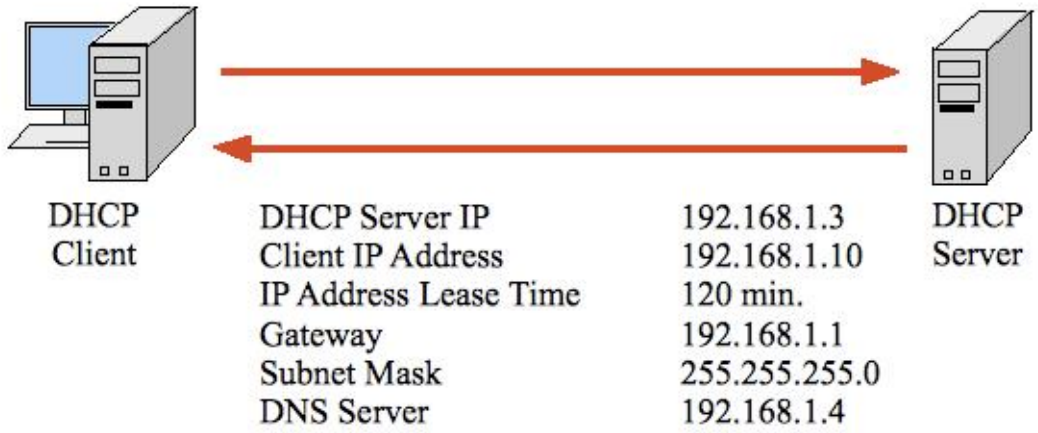


I1223: Example DHCP Interaction

From a DHCP perspective, there are only two kinds of entities: DHCP Clients (network devices or hosts) and DHCP Servers.

DHCP Clients

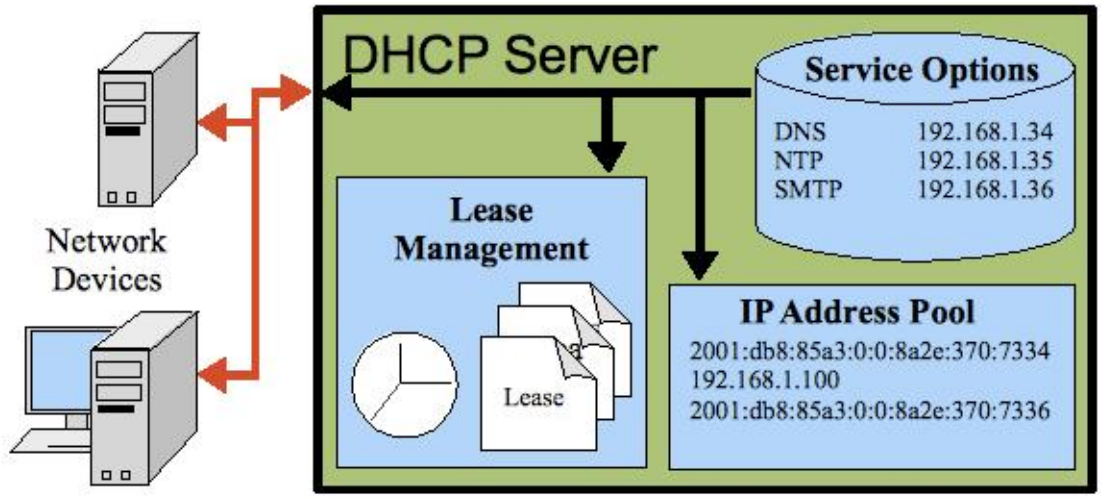
DHCP clients, sometimes referred to as network devices or hosts, use the network to contact the DHCP Servers to obtain an IP address and the configuration parameters required to use that connection. Once configured, the DHCP client then obtains the IP addresses of the network services (i.e., **Domain Name System** [**DNS**] server, **Network Time Protocol** [**NTP**] server, etc.) required to accomplish necessary tasks. All IP addresses a DHCP server provides are only leased to the DHCP client; the client needs to be able to recover when the DHCP server revokes the IP addresses the server allocated to the client.



I1224: Example DHCP Interaction

DHCP Servers

DHCP servers dynamically allocate IP addresses to DHCP clients dynamically and manage the leases of those addresses. In addition, the DHCP server can provide the DHCP client with the IP addresses of the various network services available on the network the DHCP Server manages. When leases expire, the DHCP Server attempts to reallocate the previous address to the same client. If the client is registered in the Domain Name System, DHCP will register any new addresses back to the DNS Server.



I1225: DHCP Server

Guidance

- G1598: Allow dynamic **Domain Name System** (**DNS**) updates to the Node's internal DNS service by local **Dynamic Host Configuration Protocol** (**DHCP**) **server(s)**.

- G1610: Configure the **Dynamic Host Configuration Protocol** (**DHCP**) services to assign **multicast** addresses.
- G1601: Use configurable **routers** to provide dynamic **Internet Protocol** (**IP**) address management using the **Dynamic Host Configuration Protocol** (**DHCP**).

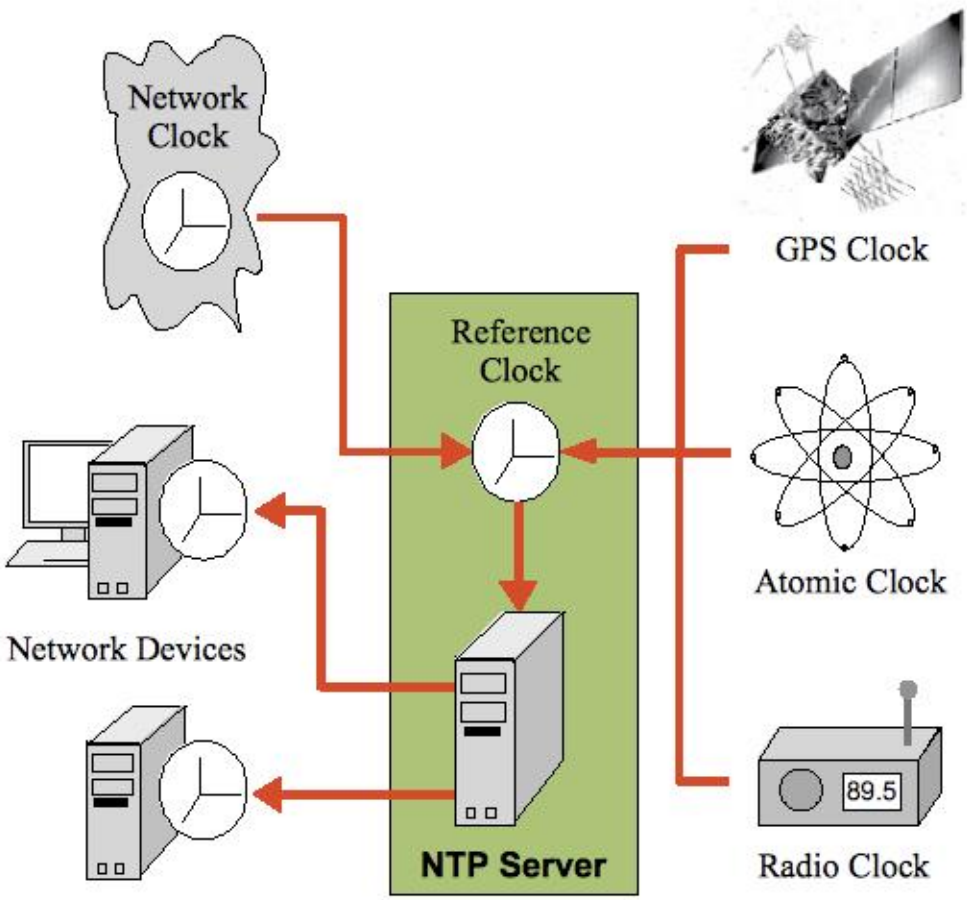
P1144: Network Time Service

Net-centric operations and security depend on date and time synchronization. Many **protocols** rely upon synchronized time to function properly, particularly security protocols. Mission **Component** logic and the usefulness of data can also suffer if there is not a common understanding and synchronization of time across the **enterprise**.

The most important and widely-used protocol for distributing and synchronizing time is the **Network Time Protocol** (**NTP**), though other less-popular or outdated time protocols remain in use.

To enable time synchronization, an NTP server reads the actual time from a reference clock and distributes this information to its clients using a computer network. The time server may be a local network time server or an internet time server. The time reference for a time server could be another time server on the network or the Internet, a connected radio clock or an atomic clock. The most common true time source is a **Global Positioning System** (**GPS**) or GPS master clock. Time servers are sometimes multi-purpose network servers, dedicated network servers, or dedicated devices. All a dedicated time server does is provide accurate time.

As an example, the U.S. Naval Observatory [http://www.usno.navy.mil] provides ***Stratum 1*** or top-level time service to Continental U.S. (CONUS) Nodes from servers at tick.usno.navy.mil and tock.usno.navy.mil. Stratum 1 time servers act as "wholesale" sources and supply time synchronization data to more local Stratum 2 "retail" time servers, which in turn provide time services to individual local systems.



I1222: Network Time Service

Guidance

- G1604: Use configurable **routers** to provide time synchronization services using **Network Time Protocol** (**NTP**).
- G1608: Obtain reference time from a standard globally synchronized time source.

- G1609: Arrange for a backup time source.

P1355: Application Layer Protocols

Internet Protocol (**IP**) networking originally developed as an environment supporting reliable transfer of digital data among a community of users. The transport infrastructure does not categorize **services**, because from the transport viewpoint it does not matter; services and Internet Engineering Task Force (IETF) "STD 66" (RFC 3986, *Uniform Resource Identifier (URI): General Syntax*) service authorities (such as **HTTP** for the Web, **FTP** for file transfer, and **SMTP** for e-mail) are just ports and associated service protocols. However, the categorization of a number of such services uses their transport port and protocol due to transport performance (**QoS**) and security reasons as well as IETF governance of many of the standards.

The user community rapidly found uses best achieved by a special protocol or protocol set that they could share in common. Some of these application layer protocols are in the following subsection.

Widely-Employed Application Layer Protocols

The Internet Protocol suite includes many application layer protocols that represent a wide variety of applications, including the following:

- ***File Transfer Protocol (FTP)*** is a network protocol used to transfer data from one computer to another through a network such as the Internet. FTP supports exchanging and manipulating files over a TCP computer network. A FTP client may connect to an FTP server to manipulate files on that server. There are many FTP client and server programs available for different operating systems, making FTP a popular choice for exchanging files independent of the operating systems involved.
- ***Simple Network Management Protocol (SNMP)*** forms part of the Internet Protocol suite as defined by the Internet Engineering Task Force. Network management systems use SNMP to monitor network-attached devices for conditions that warrant administrative attention. SNMP consists of a set of standards for network management, including an Application Layer protocol, a database schema, and a set of data objects.
- ***Telnet*** (a contraction of ***Tel***ecommunication ***net***work) is a network protocol used on **Internet** or **local area network** (**LAN**) connections. The term telnet also refers to software which implements the client part of the protocol. Telnet clients are available for virtually all platforms. Most network equipment and operating systems with a TCP/IP stack support some kind of Telnet service server for their remote configuration.
- ***X Windows*** is a windowing system that implements the X display protocol and provides windowing on bitmap displays. It provides the standard toolkit and protocol with which to build graphical user interfaces (GUIs) on most Unix-like operating systems and OpenVMS. The X Windows system has been ported to many other contemporary general purpose operating systems.
- ***Network File System (NFS)*** is a network file system protocol which allows a user on a client computer to access files over a network as easily as if the network devices were attached to its local disks.
- ***Simple Mail Transfer Protocol (SMTP)*** is a standard for electronic mail (e-mail) transmissions across the Internet. While electronic mail server software uses SMTP to send and receive mail messages, user-level client mail applications typically only use SMTP for sending messages to a mail server for relaying. For receiving messages, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) to access their mail box accounts on a mail server.
- ***Hypertext Transfer Protocol (HTTP)*** is an application-level protocol for distributed, collaborative, hypermedia information systems. HTTP is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers.
- ***Secure Shell (SSH)*** is a network protocol that allows data exchange using a secure channel between two networked devices. SSH was designed as a replacement for TELNET and other insecure remote shells which sent information, notably passwords, in plaintext, leaving them open to interception.
- ***Session Initiation Protocol (SIP)*** is a signalling protocol, widely used for setting up and tearing down multimedia communication sessions such as voice and video calls over the Internet. Other feasible application examples include video conferencing, streaming multimedia distribution, instant messaging, presence information and online games. The protocol can be used for creating, modifying and terminating two-party

(unicast) or multiparty (**multicast**) sessions consisting of one or several media streams. The modification can involve changing addresses or ports, inviting more participants, adding or deleting media streams, etc.

P1141: Mobility

There have been significant advances in **Transmission Control Protocol/Internet Protocol** (**TCP/IP**) connectivity to mobile **Nodes**, such as airplanes, ships, and battlefield units; however, some significant challenges remain. In particular, it is unclear to what extent mobile Nodes can utilize **Enterprise Services**, particularly the DISA **Core Enterprise Services** (**CES**), directly. The characteristics of the link are likely to be extremely variable, including high frequency of topology changes, intermittent connectivity, higher than typical packet loss, low bandwidth, or high latency. Such characteristics are generally problematic for anything but the simplest of enterprise services. Components that use these services need to adapt in real-time to the presence or absence of the service and to the potentially intermittent performance of enterprise services. Consequently, these components must be able to handle the failover and recover from enterprise service errors and gaps.

Managers of mobile Nodes that rely on the **Internet Protocol** (**IP**) for inter-Node communication should engage with the DISA **Net-Centric Enterprise Services** (**NCES**) Program Office [R1259] to explore approaches for mobile use of the CES services. Alternatives might include development of specialized **Software Developers Kits** (**SDKs**) that implement the required adaptive behavior or use of service **proxies** within the Node that could failover gracefully.

Many of the transport elements listed above may require extensions to account for the Node's intended mobile environment. For example, today's commercial routing protocols are not intended for the extent of dynamic and mobile behavior encountered in tactical military environments.

Another example is that **TCP** performance over satellite links is generally poor due to delays and blockages inherent to satellite links. Consider TCP extensions and other transport protocols developed to mitigate this risk for high bandwidth, high latency satellite communications.

Mobile IP is a standard that allows users with mobile devices whose IP addresses are associated with one network to stay connected when moving to a network with a different IP address. When a user leaves the network with which his device is associated (home network) and enters the domain of a foreign network, the foreign network uses the Mobile IP protocol to inform the home network of a care-of address to which to send all packets for the user's device.

Nodes can be mobile or deployable as well as fixed. Mobile networks, by their very nature, are untethered and usually reliant upon radio frequency (RF) transmissions. An inherent challenge to address is that of ensuring uninterrupted **Global Information Grid** (**GIG**) interoperability as the underlying network changes dynamically.

> ***Note:*** *A goal of mobile or deployable Nodes is that they can plug into different locations in the GIG without loss of interoperability.*

Mobile IPv4

A mobile node can have two addresses:

- a permanent home address
- a care-of address associated with the network the mobile node is visiting

There are two kinds of entities in Mobile IP:

- a home agent stores information about mobile nodes whose permanent address is in the home agent's network
- a foreign agent stores information about mobile nodes visiting its network; foreign agents also advertise care-of addresses which Mobile IP uses

A node wanting to communicate with the mobile node sends packets to the home address of the mobile node. The home agent intercepts these packets and, using a table, tunnels the packets to the mobile node's care-of address with a new IP header while preserving the original IP header. Decapsulation at the end of the tunnel removes the added IP header from the packets prior to delivery to the mobile node.

When acting as a sender, a mobile node simply sends packets directly to the other communicating node through the foreign agent.

Mobile IPv6

A key benefit of Mobile IPv6 as opposed to Mobile IPv4 is that even though the mobile node changes locations and addresses, the existing connections through which the mobile node is communicating are maintained. To accomplish this, connections to mobile nodes are with a specific address always assigned to the mobile node and through which the mobile node is always reachable. Mobile IPv6 provides Transport layer connection survivability when a node moves from one link to another.

Best Practices

- BP1594: Examine the use of **Transmission Control Protocol** (**TCP**) extensions and other transport protocols that have been designed to mitigate risk for high bandwidth, high latency satellite communications.

P1356: Traffic Management

Network traffic management uses the principles of Traffic Engineering and **Quality of Service** (**QoS**) to optimize the network by dynamically analyzing, predicting and regulating the behavior of the network in transmitting data. Although traffic engineering originated in the telecommunications industry, the principles have been applied successfully to all kinds of communications networks including **local area networks** (**LANs**), wide area networks (WANs), cellular telephone networks and the **Internet**.

A major objective of traffic management is to optimize network performance to meet a wide variety of mission objectives. To accomplish this, traffic management must maximize the timely transport of traffic while simultaneously minimizing traffic loss, traffic exposure to compromise (particularly denial of service attacks) and operations/maintenance costs

Striking this balance between effective, secure and efficient Transport requires engineering embedded sensor and control points and engineering enterprise operations support systems that integrate network situation information and coordinate performance management operations

Good traffic management applied to network infrastructure enhances performance metrics, such as bandwidth, delay and interference, by defining administrative policies in accordance with commanders' intentions that govern traffic admission, aggregation, response to congestion, error handling, etc. Poor choices in such policies result in traffic delay, loss, and interference; however, good choices result in timely, responsive, robust information flows.

A way to avoid congestion, for example, is matching capacity to usage or usage to capacity. The matching process may occur either before access, as part of planning, or during usage spikes/troughs as an adaptive mechanism. Planning allows network service consumers to request a baseline service contract with the service provider. Specify the service consumer's requirements for bandwidth and other performance metrics as part of a **Service Level Agreement** (**SLA**). The network service determines if there is enough bandwidth available to fulfill the request. If there is enough capacity, the bandwidth is allocated to the consumer. If there is not enough capacity, the service consumer is rejected or capacity is added to the network.

In an ideal world, with proper network planning, networks should never be congested or suffer interference. However, the reality is that networks do have congestion either from fulfilling unplanned network service requests (i.e., load) or as a result of a degraded network. Congestion is only one performance tradeoff failure; another involves interference and noise which interact with congestion. Interference causes congestion due to error correction and retransmission, and congestion causes interference due to interactions inside of shared resources. The network traffic can respond to these conditions through various traffic engineering principles such as restricting or buffering network capacity.

Quality of service is a defined level of performance that adapts to the environment in which it is operating. The user of the information may be request the required QoS. The level of QoS provided is based on the request, the available capabilities of the provider, and the priority of the user.

Class of Service (CoS) is a queuing discipline. The CoS algorithm compares fields of packets or CoS tags to classify packets in different priority queues by grouping similar types of traffic and treating each type as a class with its own level of service. Class of service is simpler to manage that quality of service. Class of service is often more coarse-grained in traffic control where quality of service is more fine-grained.

The two taken together are a means for the user to specify the level of performance that he desires and the network engineer to attempt to provide that service. QoS is derived from a capability in Asynchronous Transfer Mode (ATM) where bandwidth is allocated and QoS can be guaranteed. QoS in IP networks is not guaranteed. It is an attempt by the IP network to provide service similar to ATM service.

Detailed Perspectives

The following perspectives provide more detailed information.

- Planning Network Services [P1357]
- Architectural Approaches to Traffic Management [P1358]
- Traffic Engineering [P1359]

P1357: Planning Network Services

Network planning is essential for meeting a desired network level of service. Planning can be static, off-line well in advance of the actual usage, or it can be dynamic in response to service consumer's requests. The network service balances the consumer's resource request against the available network resources and, if possible, reserves the network resources for the consumer.

To accomplish the planning and administration of the network, traffic engineering abstracts the network as a service governed by a service contract. As with most contracts, there are two independent types of parties (with at least one of each type) involved: service provider and service consumer. **Service Level Agreement** (**SLA**) parameters define the terms and conditions of a network service. The SLA parameters capture the levels of availability, serviceability, performance, operation or other service attributes as reflected in performance metrics. The SLA parameters are expressed as one or more Service Level Objectives (SLOs) which must be measurable, repeatable, attainable, controllable within measured bounds, and mutually acceptable.

Network **Quality of Service** (**QoS**) provides an assessment of "excellence" of the network service. The assessment is for each of the SLA parameters. Each SLA parameter assessment represents an aggregate of the compliance measures for the individual SLOs.

SLA Parameter	Explanation	SLO Example
Availability	Constraints on when the service can be used by the provider or when it is needed by the consumer	Network shall be available 99.9% of the time in delivering traffic to and from IP endpoints
Accessibility	Enablers or barriers to use of a service as specified by the provider or for facilities for overcoming the barrier by the consumer	Network shall support IPv4 and IPv6 traffic
Performance	Sustainable rate of providing the service or the demand for capacity from the consumer	Network latency shall be 40 milliseconds or less between IP endpoints
Compliance	Assurance of the quality of the product provided by the producer or required by the consumer	Network shall comply with IPv6
Security	Risk to the provider in servicing consumer or to the consumer in using the provider's service	Network shall support a minimum of a 1024-bit cryptographic keys
Efficiency	Cost of servicing a consumers request or using the producer's product	Networks shall support a network packet sizes from 512 to 16,384 bytes
Reliability	Assurance consistency of the product by the producer or the expectation of consistency of the product by the consumer	Network IP Packet loss shall not exceed 0.1% based on the arithmetic mean of the aggregate monthly measurement between IP endpoints
Provenance	Assurance of the origin and history of the product by the producer or the expectation of the origin and history of the product by the consumer	Network traffic shall only be on wired networks

P1358: Architectural Approaches to Traffic Management

The following standards-based **Quality of Service** (**QoS**) approaches to Traffic Management are two examples of those used both on commercial enterprise **intranets** and in the DoD. The Differentiated Services (DiffServ) architecture enables course-grain deconfliction and priority labeling of traffic in accordance with a business model or commander's intent. The Integrated Services (IntServ) architecture enables fine-grain traffic deconfliction and prioritization, but the extra control comes at a price: higher operational costs, greater network operational complexity, and overall network brittleness.

Differentiated Services

DiffServ is a networking architecture that specifies a simple, scalable, coarse-grained mechanism for classifying network traffic, managing network traffic, and providing Quality of Service (QoS) guarantees on modern IP networks. As such, it allows senior commanders to prioritize traffic over shared infrastructure according to technology and mission needs by separating it into classes and trading-off resource allocation according to class. DiffServ can, for example, provide low-latency, guaranteed service (GS) to critical network traffic such as voice or video while providing simple best-effort traffic guarantees to non-critical services such as Web traffic or file transfers. DiffServ exhibits good scaling properties. However, in the absence of additional conditioning mechanisms, DiffServ provides only preferential, differentiated levels of service and not guarantees.

Traffic flows into a DiffServ policy domain through its ingress boundary router, which then classifies and marks it with the appropriate DiffServ Code Point (DSCP) marking. From that ingress router on, the traffic is routed along its path through internal routers, which condition the traffic stream in accordance with the policies specified by the Traffic Conditioning Agreement (TCA) associated with that DSCP marking. All traffic leaving a Diffserv domain does so through an egress boundary router, which acts as the limit of the policy and the commander's span of control. For end to end traffic policy compliance, the ultimate client endpoint router should also be the egress router.

The following Internet Engineering Task Force (IETF) Requests for Comments (RFCs) provide additional information:

- RFC 2474, Standards Track, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, December 1998
- RFC 2475, Informational, *An Architecture for Differentiated Service*, December 1998
- RFC 4124, Proposed Standard, Protocol *Extensions for Support of Diffserv-aware MPLS Traffic Engineering*, Jun 2005.
- RFC 4125, Experimental, *Maximum Allocation Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*, Jun 2005.
- RFC 4594, Informational, *Configuration Guidelines for DiffServ Service Classes*, Aug 2006.
- RFC 3270, Proposed Standard, *Multi-Protocol Label Switching (MPLS) Support of Differentiated Services*, May 2002.

Integrated Services

IntServ is an architecture that specifies the elements to guarantee quality of service (QoS) on networks. IntServ can, for example, allow video and sound to reach the receiver without interruption. IntServ specifies a fine-grained QoS system, which is often contrasted with a DiffServ coarse-grained control system. The idea of IntServ is that every router in the system implements IntServ, and every application that requires some kind of guarantees has to make an individual reservation. "Flow Specs" describe what the reservation is for, while "RSVP" (in this usage, Resource ReSerVation Protocol) is the underlying mechanism to signal it across the network.

IntServ is based on a network traffic engineering model that primarily serves the real-time flow of **IP** packets along a network path of IP nodes between two endpoints (i.e., end-to-end). IntServ accomplishes this by reserving a portion of the network bandwidth to the flow of IP packets along the designated network path. The packets flowing within the reserved bandwidth behave deterministically along the path. Packets that are not apportioned to a dedicated portion of the bandwidth remain highly non-deterministic. In other words, the packets under the control of IntServe flow under a reserved apportionment of the bandwidth. The IETF first proposed the IntServ model in

1993 as RFC 1663 primarily to support real-time teleconferencing, remote seminars, telescience and distributed simulation services.

In an IntServe architecture, a data flow starts with a request from a potential consumer (i.e., requestor) of a data stream (i.e., broadcast). How the consumer discovers the source of the broadcast is outside the scope IntServe. The consumer makes a reservation request to its router. The router then passes the request up stream to all the routers in the path to the broadcaster. If there are multiple consumers of the broadcast, the reservations are merged as they move upstream to help reduce network traffic. As the router can service the reservation, the broadcast starts to flow from the broadcaster to the consumer. If a router is already servicing a broadcast request at or above the requested data rate from another consumer, the reservation request does not need to go up stream any further and the broadcast can start flowing to the consumer from that router.

Note: *Broadcasts can be separated into various layers, with each layer representing a particular quality range. For example, a 20Kbps low quality audio layer may be encoded separately from the high quality enhancement of the audio. Additionally, the video aspect of the broadcast can be encoded into yet more layers.*

Hosts on the **Internet** use the Resource Reservation Protocol to request a QoS level on the network on behalf of an application data flow. Routers use RSVP to deliver QoS requests to other routers along the path(s) of the data flow. The impacts of using RSVP over the black core must be understood and accounted for as more information about the black core becomes available.

The following IETF RFCs provide additional information:

- RFC 2205, Proposed Standard, *Resource ReSerVation Protocol RSVP -- Version 1 Functional Specification*, September 1997.
- RFC 2207, Proposed Standard, *RSVP Extensions for IPSEC Data Flows*, September 1997.
- RFC 2998, Informational. *A Framework for Integrated Services Operation over Diffserv Networks*, Nov. 2000.
- RFC 1633, Informational, *Integrated Services in the Internet Architecture: an Overview*, Jun 1994,

QoS-Based Routing

QoS-based routing is a mechanism under which paths for flows are determined based on some knowledge of resource availability in the network as well as the QoS requirement of flows. These protocols search for routes with sufficient resources for the QoS requirements. QoS-based routing also has potential to address tactical edge environments; however, the overhead of QoS routing protocols is very high for bandwidth-limited mobile ad hoc networks (MANETs).

The following IETF RFCs provide additional information:

- RFC 2386, Informational A Framework for QoS-based Routing in the Internet, Aug 1998.
- RFC 2676, Experimental QoS Routing Mechanisms and OSPF Extensions, Aug. 1999.
- RFC 3583, Informational Requirements of a Quality of Service (QoS) Solution for Mobile IP, Sep 2003.

P1359: Traffic Engineering

Traffic engineering is a method of optimizing the performance of a network by dynamically analyzing, predicting and regulating the behavior of data transmitted over that network. Traffic engineering uses statistical techniques such as queuing theory to predict and engineer the behavior of telecommunications networks such as telephone networks or the **Internet**. The crucial observation in traffic engineering is that in large systems the law of large numbers can help make the aggregate properties of a system over a long period of time much more predictable than the behavior of individual parts of the system. The queueing theory originally developed for circuit-switched networks is applicable to packet-switched networks.

Traffic Classification

Packet classifiers select **Internet Protocol** (**IP**) packets in a traffic stream based upon the content of some portion of the packet header. In essence, classifiers "steer" packets matching some specified rule to an element of a traffic conditioner for further processing. Classifiers must be configured by some management procedure in accordance with the appropriate Traffic Conditioning Agreement (TCA).

In the Differentiated Services (DiffServ) architecture, two basic types of classifiers exist. The first is a multifield (MF) classifier, which examines multiple fields in the IP datagram header to determine the service class to which a packet belongs. The second is a behavior aggregate (BA) classifier, which examines a single field in an IP datagram header and assigns the packet to a service class based on what it finds.

Behavior Aggregate (BA) Classifier

The BA classifier classifies IP packets based solely on the Differntiated Services Code Point (DSCP). Specific DSCP values are used as the selector for per-hop behavior (PHB).

Multi-Field (MF) Classifier

The MF classifier is used when the BA classifier is insufficient to classify a packet. The MF classifier selects IP packets based on the value of a combination of one or more IP header fields (i.e., source address, destination address, Differntiated Services field, protocol ID, source port, destination port numbers, and DSCP).

Note: *Sometimes the packets are fragmented from each other upstream in the packet stream. When an MF classifier uses the contents of transport-layer header fields, it may not consistently classify subsequent packet fragments. A possible solution is to maintain a fragmentation state; however, this is not a general solution due to the possibility of upstream fragment re-ordering or divergent routing paths.*

Traffic Conditioning

Traffic conditioning can involve the metering, shaping, policing and/or re-marking of packets to ensure that traffic conforms to the rules specified in the Traffic Conditioning Agreement and in accordance with the domain's service provisioning policy. The extent of traffic conditioning required is dependent on the specifics of the service offering. Conditioning might be simple DSCP re-marking or very complex policing and shaping operations.

Classifiers select a traffic stream and then direct packets to a logical instance of a traffic conditioner. A meter might measure the traffic stream against a traffic profile. The state of the meter with respect to a particular packet (e.g., whether it is in-profile or out-of-profile) may be part of the traffic marking, dropping, or shaping actions.

Note: *A traffic conditioner may not necessarily contain all four conditioning operations (metering, shaping, policing, re-marking). For example, if there is no traffic profile in effect, packets may only be subject to the classifier and marker operations.*

Representative traffic engineering building blocks follow.

Bandwidth Management

Bandwidth management is the process of measuring and controlling the communications (traffic, packets) on a network link to avoid filling the link to capacity or overfilling the link, which would result in network congestion and poor performance. More sophisticated bandwidth management techniques use a macro approach that manages traffic on a per user rather than a per application basis. This frees the network provider from having constantly to identify what clients/customers are doing and avoids some of the legal concerns and public outcry about providers dictating what customers can do. This approach acknowledges that on Internet Service Provider (ISP) type networks, "fairness" is a per client issue. By managing per client, no single user can use more bandwidth than the user's allocation, no matter what application the user may be running or how many users are on the user's endpoint.

Admission Control

Admission control is a mechanism that estimates the level of QoS that a new user session will need and whether sufficient bandwidth is available. If bandwidth is available, the session is admitted. Admission control is a network **Quality of Service** (**QoS**) procedure. Admission control determines how bandwidth and latency are allocated to streams with various requirements. An application that wishes to use the network to transport traffic with QoS must first request a connection, which involves informing the network about the characteristics of the traffic and the QoS the application requires. This information is stored in a traffic contract. The network judges whether it has enough resources available to accept the connection and then either accepts or rejects the connection request. Admission control is useful in situations where a certain number of connections (phone conversations, for example) may all share a link, while an even greater number of connections causes significant degradation in all connections to the point of making them all useless such as in congestive collapse.

Prioritization

Prioritization is a mechanism to give important network traffic precedence over unimportant network traffic. Prioritization is also called class of service (CoS) since traffic is classed into categories such as high, medium, and low (or gold, silver, and bronze, etc.), and the lower the priority, the more "drop eligible" is a packet.

Rate Limiting

Rate limiting is the process of restricting a classified packet flow or a source interface to a rate that is less than the physical rate of the port. Rate limiting enforces data rates below the physical line rate of a port for an IP interface, a classified packet flow, or a Layer 2 interface. It allows limiting the total bandwidth one class of traffic uses and making it available for other classes. Some implementations allow hierarchies of rate limits with preferential access among them.

Delay Management

Delay Management is a capability to control traffic in order to optimize or guarantee performance, low latency, and/or bandwidth by delaying packets. Delay and latency are similar terms that refer to the amount of time it takes to transmit a bit from source to destination. One way to view latency is how long a system holds on to a packet. That system may be a single device like a router, or a complete communication system including routers and links (derived from the Linctionary.com Delay, Latency, and Jitter entry, http://www.linktionary.com/d/delay.html). Traffic shapers delay some or all of the packets in a traffic stream in order to bring the stream into compliance with a traffic profile.

IP QoS manages delay of packets through a router. However, in wireless environments, such as an airborne network, the transmission time over a line-of-sight link is likely to dominate delays. In such cases, delay management through the router will be important mostly for queuing outgoing packets on the radio link.

Drop Management

Drop management is a capability to alleviate congestion by dropping packets when necessary or appropriate. Drop management includes mechanisms such as admission control (drop all traffic before queuing), pre-emption (drop all traffic henceforth), active queue management (for example Random Early Detection (RED), and Weighted RED which drops selected traffic packets. Refer to the Internet Engineering Task Force (IETF) *Recommendations on Queue Management and Congestion Avoidance in the Internet* Request for Comment (RFC 2309).

P1388: Text Conferencing

Text conferencing, sometimes called ***on-line chat*** or simply ***chat***, is a synchronous text-based communication. The common English definition of chat implies something less than serious; however, on-line chat is a very serious and effective means of communication (i.e., collaborating) that can convey important, formal dialog between the participants. Information that flows between participants is not limited to simple text but can covey complex constructs that reflect information, knowledge, understanding and even wisdom. Recently, text communication has moved beyond human-to-human dialog and has become increasingly used to connect automated software agents to humans and other software agents

Text conferencing provides the ability to transmit plain text messages between inidividuals or groups of individuals in near-real-time. Some implimentations support structured messages that help the text conferencing infrstructure process and distribute the text as desired by the sender. Text conferencing implementations generally have the following qualities:

- Allow for the rapid dissemination of information
- Provide a history of communications useful for after action reviews or to catch up on missed messages
- Support filterable inbound message traffic
- Operate at the security level of the underlying network
- Are simple to use
- Require minimum bandwith and are easily compressed
- Reduce voice network traffic
- Overcome elctro-magnetic interferences
- Overcome line-of-sight of radio limitations
- Provide a means for finding, retrieving, and subscribing to changes in the presence status (e.g., "online" or "offline") of users

There are predominately two protocols that govern text communication: Internet Relay Chat (IRC), and Extensible Messaging and Presence Protocol (XMPP).

Internet Relay Chat (IRC)

Internet Relay Chat (IRC) is a form of near-real-time synchronous conferencing that is comprised of a network of IRC servers and IRC clients. The IRC network optimizes the routing of messages between clients by only transmitting a message once along any network link.

There are several types of software components that interact with IRC networks: ***user clients***, ***bouncers***, and ***bots***. IRC user clients simplify for human users the use of IRC messages, usually with an easy-to-use interface. IRC bouncers run on a server and act as persistent proxies for the user clients, supporting intermittent connectivity between the IRC server and the IRC user client. IRC bots often provide high-speed, automated IRC services such as registration and management. Bots can be in any number of languages since the IRC protocol acts as a standardized message based interface. Additionally, bots may execute in a user session to assist with common tasks.

Additional IRC Information Sources

- IETF RFC1459, Internet Relay Chat Protocol, May 1993
- IETF RFC2810, Internet Relay Chat: Architecture, April 2000

Extensible Messaging and Presence Protocol (XMPP)

The Extensible Messaging and Presence Protocol (XMPP) is an **eXtensible Markup Language** [**XML**] protocol for providing near-real-time synchronous text conferencing and presence information. XMPP- based text conferencing infrastructure is comprised of a network of XMPP servers and XMPP clients.

XMPP clients send XMPP XML messages to an XMPP server. The XMPP messages can be messages for other clients or commands that are to be processed by the XMPP servers. XMPP servers are tasked with maintaining the ***presence*** of XMP clients (users) on the XMPP network. As XMPP clients join and leave the XMPP network, their presence is made available to other XMPP clients that have expressed interest in those XMPP clients.

XMPP gateways can link XMPP networks to other networks such as email (**SMTP**), Internet Relay Chat (IRC), Session Initiation Protocol (SIP) for Instant Messaging and Presence Leveraging Extensions (SIMPLE), and Short Message Service (SMS) as well as other legacy networks (see Application Layer Protocols [P1355] for additional information). XMPP only defines the concept of a gateway; the implementation of the gateways is outside the scope of XMPP.

XMPP relies on the use of the Jabber Indentifier (JID) which ties the identificaion of the XMPP client (user) to a domain (i.e., <**`node@domain/resource`**>. This scheme is similar to the methods used to deliver email but it is not similar to the method used by Internet Relay Chat (IRC) which has a limit of characters and is tied to the host name. This difference in structure and size of structured identifiers used to identify users can limit interoperability of user identifiers between XMPP and IRC systems.

The current base XMPP specifications are RFC 3920 and RFC 3921 (see the additional XMPP information sources below). However, the **Internet Engineering Task Force** (**IETF**) XMPP Working Group is revising these specifications to incorporate lessons learned from current implementation challenges.

Additional XMPP Information Sources

- XMPP Standards Foundation, http://xmpp.org
- IETF RFC3920, *Extensible Messaging and Presence Protocol (XMPP): Core*, October 2004
- IETF RFC3921, *Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*, October 2004
- XEP0205, *Best Practices to Discourage Denial of Service Attacks*, Version 0.2, 10 July 2007

Best Practices

- BP1907: Use Internet Relay Chat (IRC) bots to provide network based IRC services.

P1365: Data Interchange Services

This service area supports information interchange between applications. NESI provides guidance that support this DISR Service Area. Use the following detailed perspectives for guidance related to this service area.

Detailed Perspectives

- Messaging [P1047]
- Web Services [P1078]
- CORBA [P1011]
- Data Distribution Service [P1190]
- Data [P1012]
- Net-Centric Information Engineering [P1133]
- Node Data Strategy [P1329]

P1047: Messaging

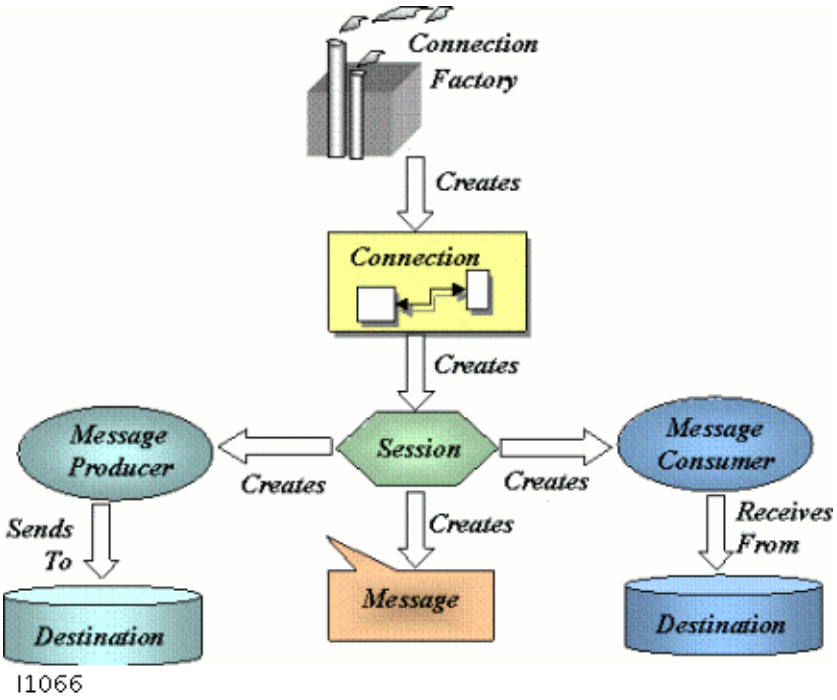
The explosion of the Internet required applications to communicate and interoperate with other applications and services. Messaging systems play an important role in enterprise applications because computers and networks are inherently unreliable and messaging systems are perfectly suited to operate in disconnected environments. They provide a reliable, secure, event-driven message-delivery communication mechanism. Unlike traditional **RPC**-based systems (**RMI** or **CORBA**), most message-oriented based systems operate peer-to-peer.

The messaging paradigm offers three major advantages:

- Allows applications to communicate asynchronously. This means the system sending the **message** does not have to wait around for a response.
- Provides more robustness and reliability; messages do not get lost if a **client** has crashed or is unavailable.
- Multiplexes messages and sends them to multiple clients.

There are other advantages such as transactional message support, message prioritization, load balancing, and firewall **tunneling**. However, these features usually depend on how the **Message-Oriented Middleware** (**MOM**) is implemented.

This diagram shows the relationship of the classes and interfaces in the **Java Message Service** (**JMS**) **API**. Developers use these classes and interfaces to create a JMS application.



P1046: Message-Oriented Middleware (MOM)

Message-oriented middleware acts as an arbitrator between incoming and outgoing **messages** to insulate producers and consumers from other producers and consumers A **MOM** typically is implemented using proprietary **protocols** and interfaces, which means that different implementations are usually incompatible. Using a single implementation of a MOM in a system typically leads to dependence on the MOM vendor for maintenance, support, and future enhancements. Maturing standards such as **Java Message Service** (**JMS**) and **SOAP Web services** are reducing vendor dependencies by standardizing message content and providing standard interfaces to the various MOM **APIs**.

Advantages

- A MOM provides a common reliable way for programs to create, send, receive, and read messages in any distributed enterprise system.
- A MOM ensures fast, reliable, asynchronous communications, guaranteed message delivery, receipt notification, and transaction control.
- A MOM increases the interoperability, portability, and flexibility of an application by allowing it to be distributed over multiple heterogeneous platforms.
- A MOM enables applications to exchange messages with remote programs without having to know on what platform or processor the other application resides.

Disadvantages

- A MOM does not help with interoperability directly, as applications need to agree on message content and format at development time.
- The current marketplace is filled with proprietary implementations of features, so moving between MOMs usually requires recoding; JMS and other standard interfaces help in this area but do not usually cover all of the vendor's extended functionality.

Features

Guaranteed message delivery	MOMs provide a message queue between interoperating processes. If the destination process is busy or offline, the message is held in a temporary storage location until it can be processed.
Asynchronous and synchronous communications	MOMs allow multitasking. Once an application sends out a message to a receiving application, the MOM allows the **client** application to handle other tasks without waiting for a response from the receiving application. Supports blocking method calls.
Transaction support	Most MOMs support transactions.
One-time, in-order delivery	MOMs guarantee that each message will be delivered once and that messages are received in the order in which they are sent.
Message routing services	MOMs support least-cost routing and can reroute around network problems.
Notification Services	MOMs provide audit trails, journaling, and notifications when messages are received.

Message models

Part 2: Traceability

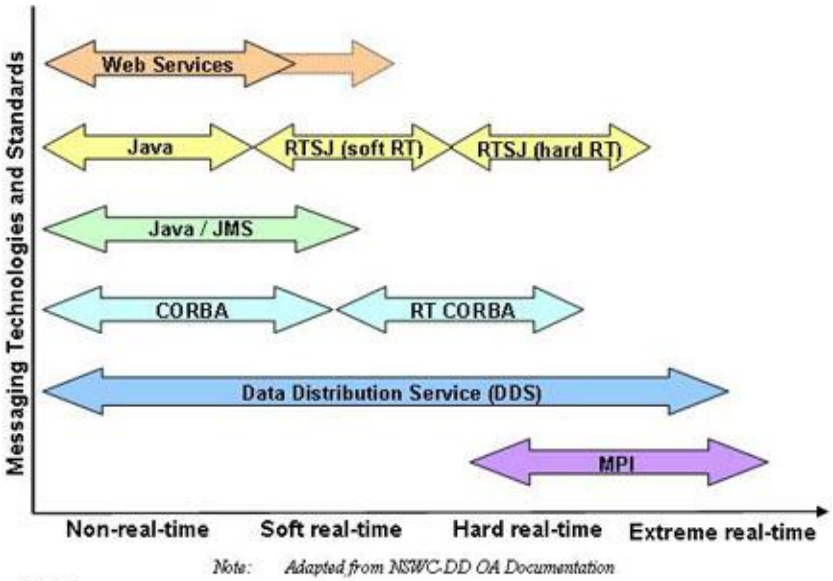
The most important aspect of a message-based communication system is the message. The most common messaging models are the following:

- Point-to-Point (p2p)
- Publish/Subscribe (pub/sub)
- Request-Reply

P1190: Data Distribution Service (DDS)

Data Distribution Service for Real-time Systems is an **Object Management Group** (OMG) specification for distributing data messages using the **Publish-Subscribe** design pattern. It defines a common application programming interface (API) that cleanly separates the data distribution functionality from the application functionality. DDS also simplifies the complexity associated with application programming by separating the details of publishing data messages from those for subscribing to data messages using a **Quality of Service** (QoS) approach. The implementation of the interface effectively creates a data distribution service that applications can access.

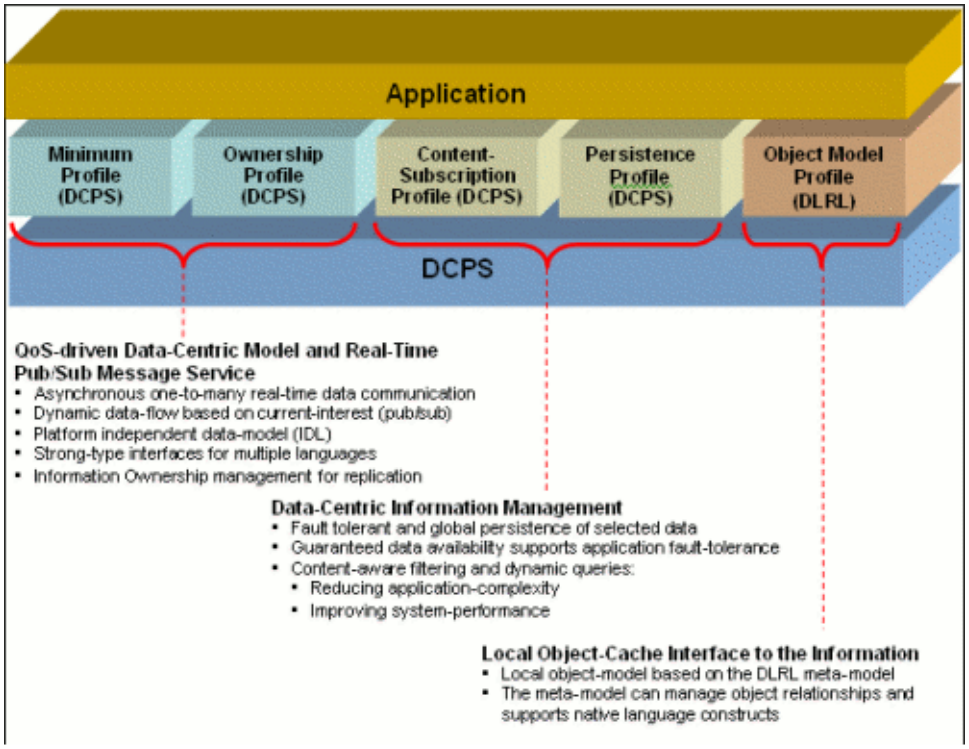
The use of QoS makes DDS especially appealing as an integration middleware in heterogeneous systems. DDS QoS allows fine-grained tuning of the properties for each information flow including the lowest level data writer and data reader. Therefore, the system can devote its resources to the more critical flows ensuring they are achievable. Also, the use of QoS combined with the inherent real-time nature of the DDS allows DDS solutions to span the complete spectrum from Enterprise (non-real-time) to hard real-time applications as shown in the following figure.



I1195

DDS Profiles

The specification divides the complexity of the full data distribution functionality into five profiles (Minimum, Ownership, Content Subscription, Persistence, and Object Model) to help applications meet their individual requirements. The applications can use any or all of the profiles to access the Data Distribution Service.



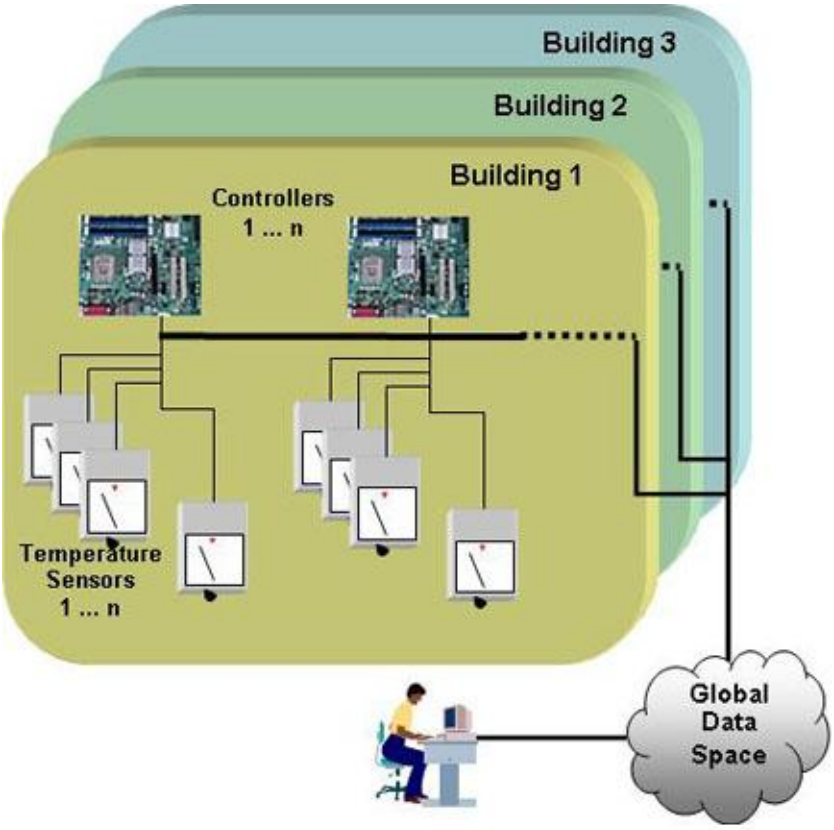
I1196

DDS Compliance Profiles

Minimum	This profile contains just the mandatory features of the **DCPS** layer. None of the optional features are included.
Ownership	This profile adds the following: • the optional setting **`EXCLUSIVE`** of the **`OWNERSHIP`** kind • support for the optional **`OWNERSHIP_STRENGTH`** policy • the ability to set a depth > 1 for the **`HISTORY`** QoS policy.
Content-Subscription	This profile adds the optional classes **`ContentFilteredTopic`**, **`QueryCondition`**, and **`MultiTopic`**. This profile also enables subscriptions by content.
Persistence	This profile adds the optional QoS Policy **`DURABILITY_SERVICE`** as well as the optional settings **`TRANSIENT`** and **`PERSISTENT`** of the **`DURABILITY`** QoS Policy kind. This profile enables saving data into either transient memory, or permanent storage so that it can survive the lifecycle of the **`DataWriter`** and system outings.
Object Model	This profile includes the **DLRL** and also includes support for the **`PRESENTATIONaccess_scope`** setting of **`GROUP`**.

Example

The following diagram depicts using a data-oriented approach to solve a typical distributed system problem. The goal in this example is to maintain the temperature in many buildings, using embedded controllers each connected to a number of sensors. Each of these sensors and control processes are connected through a transport mechanism such as Ethernet and use basic protocols such as **TCP**-**UDP**/**IP** to provide standardized communication.



I1197

To achieve data integrity and fail-over capabilities, multiple controllers and sensors are deployed in each building. Controllers within a building collaborate in the process of collecting data from the various sensors. Applications access and manipulate the data through the use of a global data space.

Data-centric technologies such as databases and Service-Oriented Architecture **Web service**-based applications can interoperate seamlessly with the embedded sensors. These technologies provide a standards-based way for external applications to get, process and manipulate real-time sensor data with out having to know the specifics of the real-time data infrastructure. Furthermore, decoupling the data from the technology that manipulates the data contributes to developing a truly data-centric application. In this example, the external access and monitoring applications can simply receive real-time updates from any sensor as well as issue commands to the various controllers via DDS, **SQL**, etc., to maintain suitable temperatures.

Data Model

For simplicity, this example will focus on the data the sensors send to their controller and how they can be distributed throughout the entire system. The first step in a data-centric approach is to describe the data format carefully in a standards-based way, either IDL or XML, and give it a **Topic** name. Topics are the element of the DDS middleware publish-subscribe standard which identify the data objects and provide the basic connection between **publishers** and **subscribers**. Subscribers (the Controllers in this example) register Topics with the middleware that they wish to receive. Publishers (the individual sensors in this example) register Topics with the middleware that they will send. If the Topics do not match, effective communication does not take place.

Topics enable one to find specific information sources when architecting a loosely coupled system; that is, one which does not know a priori how many sensors or controllers there are going to be or where they all are. The Controller can simply subscribe to **`TempSensor`**, the Topic's name, and receive all the sensor updates for that building. Similarly, a sensor does not need to know if it is sending its data to one or multiple Controllers or even an external data store.

Specification of the Topic's name is a key element in a **data-centric** approach to creating open **real-time systems**. One could name each sensor's Topic based on its unique location in the building, **`Floor12Room3Sensor14`** for example, but the Controller would then need to be configured every time a sensor is added or removed from the system. Topics (name and type) define the standard interface for the distributed system; chose them appropriately.

Data Type

Specification of the Topic's data type is equally important as the Topic's name. DDS specifies the use of a subset of the **Interface Definition Language** (IDL) for specifying a Topic's data type.

> ***Note:*** *IDL readily maps to XML and SQL semantics.*

```
struct SensorData
{
  long    id; //@key
  float   temp;
};
```

I1198

In the definition of the Topic's type, chose one or more data elements to be a ***Key***. Keys provide scalability and the communication infrastructure can use the key to sort and order data from many sensors. In this example, without keys, one would need to create individual Topics for each sensor. Topic names for these topics might be **`Sensor_1`**, **`Sensor_2`**, and so on. Therefore, even though each Topic is comprised of the same data type, there would still be multiple Topics. With keys, there is only one topic, **`TempSensor`**, used to report temperatures.

New sensors can be added without creating a new Topic. The publishing application would just need to set a new id when it was ready to publish. An application can also have a situation where there are multiple publishers of the same Topic with the same key defined. This enables the application to provide redundancy. Per this example, two sensors in the same room using the same Key value will measure the same piece of information. Managing the redundancy, should one or both sensors report to the controller, is accomplished though Quality-of-Service (QoS).

Domains and Partitions

A **Domain** is the basic DDS construct used to bind individual **publications** and **subscriptions** together for communication. A distributed application can elect to use single or multiple DDS Domains for its data-centric communications. A Partition is a way to separate Topics logically within a DDS Domain.

In the context of the example, Partitions can group sensors on different floors. For example, to divide the building into different zones where each zone is controlled by a dedicated Controller, the Sensor and Controller could set the Partition to **`Floor 1`** and **`Floor 1-6`**, respectively. The Controller will receive data from all Sensors on Floors 1 through 6. Using Partitions makes it easy to group which Sensors are ***hooked*** to a Controller and a Controller can take over a different zone by changing or adding to its Partition list.

In the example, different buildings map to different DDS Domains. Domains isolate communication, promote scalability and segregate different classifications of data.

Quality of Service

The following briefly details how one might leverage a few of the DDS QoS Policies for this example.

Ownership

The Ownership QoS specifies whether or not multiple publishers can update the same data object and is how to achieve fault-tolerance using DDS.

Returning to the example, having multiple sensors in the same room and only wanting to get data from the primary (as long as it is functioning), then the Ownership QoS policy is set to Exclusive, stating that only one sensor can update that keyed value. Setting the Ownership QoS value to Shared indicates that there can be multiple sensors in the same room all reporting the same piece of keyed data. In this case the Controller would get all updates from all sensors and treat the values as the same measurement.

Durability

The Durability QoS specifies whether past samples of data will be available to newly joining subscribers.

Considering the example, if a Controller were to reboot, rather than require all sensors to resend their data, or require the data to be sent at a periodic rate in case the systems reboots, one simply gets the latest published value for every attached sensor. This effectively decouples the system in time and provides a high degree of data integrity.

History

History specifies how many data samples are stored for later delivery.

In the case of the example, a rebooted controller may want the last 5 samples from its sensors, so that it can make sure that readings are consistent.

Reliability

The Reliability QoS may be set on a per Topic basis and informs the middleware that the Subscription should receive all data (no missed samples) from a Publication even over non-reliable transports. Generally for periodic publications Reliability doesn't need to be set, since it can just get the updated value one sample period later. Although periodic sensor data doesn't need to be delivered reliably, synchronization commands between Controllers in this example could be.

Summary

This simply stated example is surprisingly complex, containing many elements of real-time messaging, data integrity and failover capabilities, integration with databases, web services, as well as scalability and modularity concerns while remaining data-centric.

Detailed Perspectives

- Decoupling Using DDS and Publish-Subscribe [P1191]
- DDS Quality of Service (QoS) [P1192]
- DDS Data-Centric Publish-Subscribe (DCPS) [P1193]
- DDS Data Local Reconstruction Layer (DLRL) [P1197]

P1191: Decoupling Using DDS and Publish-Subscribe

A fundamental tenet of data-centricity and DDS is the decoupling between information providers and consumers. The decoupling is conceptually anonymous in that the producers do not need to know who the consumers are, and similarly the consumers do not need to know who the producers are. They are in fact each communicating independently using the DDS **Domain** (i.e., **Global Data Space**). Persistence services in the Global Data Space allow data written by an application to be available to late joining applications, even if the original application is no longer present.
While communications can precede anonymously, DDS does offer the means for an application to detect its communication partner. A ***Writer*** can see who the matched Readers are, and similarly a ***Reader*** can identify the matched Writers. If so requested, the application is given notification of new matches and can even "veto" specific Readers or Writers.

Decoupling and anonymity is accomplished using the publish-subscribe paradigm. Applications that want to provide information indicate their intent to publish by creating a **DataWriter** and specifying the offered **Quality of Service** (QoS) and a **Listener**. Applications that want to access information indicate their intent to subscribe by creating a **DataReader** and specifying the requested QoS and a Listener.

Publishers are matched with **subscribers** by DDS using the **Topic** and the QoS, and DDS automatically sets up the needed communication paths and resources such that information (data updates) can flow directly with the highest possible performance. **Listeners** are used to indicate to the application that certain events of interest have taken place, such as the arrival of new information for **`DataReaders`**, violations in the QoS contracts, matching of new Publishers/Subscribers or other middleware-observed events.

QoS contracts provide the means for applications/components to remain modular and independent from each other while at the same time having some control over how the information is provided or delivered. For example, a reading application may have some minimum requirements regarding reliability, ordering, coherence, or frequencies of updates, and a writing application may have some resource limits with regards to how much history it can maintain or how many readers it can handle. The QoS contract can specify these requirements and DDS checks and monitors them. In addition QoS can configure resources, message priorities, history, etc. The ability to fine-tune separately the behavior of each **`DataWriter`** and **`DataReader`** is one of the reasons why DDS can span the range from real-time to near-time to enterprise systems.

Guidance

- G1807: Check the return values of **Data Distribution Service** (DDS) functions.
- G1802: Catch **Data Distribution Service** (DDS) events.
- G1809: Handle all **Data Distribution Service** (**DDS**) events using one of the **subscriber access APIs**.
- G1810: Use **data models** to document the data contained within the **Data Distribution Service** (DDS) **Data-Centric Publish Subscribe** (DCPS).

Best Practices

- BP1811: Isolate all use of vendor specific extensions to the **Data Distribution Service** (DDS).
- BP1825: Use the **`ignore_participant`** operation on the **DomainParticipant** to deny access to another DomainParticipant trying to join a **Data Distribution Service** (DDS) **Domain**.
- BP1827: Use the **`ignore_publication`** and **`ignore_subscription`** on the **DomainParticipant** to deny access to a **Data Distribution Service** (DDS) **Topic** by a specific **DataWriter** or **DataReader**.
- BP1830: Use the **Data Distribution Service** (DDS) Content Profile to tailor subscription message data.
- BP1831: Use the **Data Distribution Service** (DDS) Persistence Profile to ensure durable data delivery.

P1192: DDS Quality of Service

Quality of Service (QoS) is a general concept that specifies the behavior of a service. Programming service behavior by means of QoS settings offers the advantage that the application developer only indicates what is wanted rather than how to achieve the specific QoS. Generally speaking, QoS is comprised of several QoS policies. Each QoS policy is then an independent description that associates a name with a value. Describing QoS by means of a list of independent QoS policies gives rise to more flexibility.

> ***Note:*** *As Service-Oriented Architecture (SOA) systems evolve and become richer in the number of publishers and subscribers supported with time, the use of well defined and specific QoS parameters becomes essential in managing the complexity of the system and the loosely coupled nature of the services.*

Data-centric communication using DDS provides the ability to specify various parameters like the rate of publication, rate of subscription, how long the data is valid, and many others. These QoS parameters allow system designers to construct a distributed application based on the requirements for, and availability of, each specific piece of data. A data-centric environment allows a communication mechanism that is custom tailored to the distributed application's specific requirements yet remains a loosely coupled design and architecture.

The ability to set QoS on a per-entity basis is a significant capability provided by DDS. Being able to specify different QoS parameters for each **Topic**, **Publisher** or **Subscriber** gives developers many options when designing their systems. Through the combination of these parameters, a system architect can construct a distributed application to address an entire range of requirements, from simple communication patterns to complex data interactions.

Guidance

- G1771: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of a **publisher**.
- G1801: Explicitly define a **Topic Quality of Service** (QoS) for each **Data Distribution Service** (DDS) Topic within a DDS **Domain**.
- G1803: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe real-time messaging criteria for **Publishers**.
- G1804: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe **DataWriter**.
- G1805: Explicitly define the **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of the **Subscriber**.
- G1806: Explicitly define the Request-Offered **Data Distribution Service** (DDS) **Quality of Service** (QoS) Policies to describe the behavior of the **DataReader**.
- G1808: Handle all **Data Distribution Service** (DDS) **Quality of Service** (QoS) contract violations using one of the **Subscriber access APIs**.

Best Practices

- BP1812: Use the **`RELIABILITY`** **Quality of Service** (QoS) kind **`BEST_EFFORT`** for **Data Distribution Service** (DDS) **Topics** that are written frequently where missing an update is not important because new updates occur soon thereafter.
- BP1813: Use the **`RELIABILITY`** **Quality of Service** (QoS) kind **`RELIABLE`** for **Data Distribution Service** (DDS) **Topics** written sporadically or where it is important that the current data in the Topic is received reliably.
- BP1814: Use the **`DEADLINE`** **Quality of Service** (QoS) to for **Data Distribution Service** (DDS) **DataWriters** for which data is published at a constant rate.
- BP1815: Use the **`DEADLINE`** **Quality of Service** (QoS) for **Data Distribution Service** (DDS) **DataReaders** that expect data to be sent to them at a constant rate.
- BP1816: Use the **`LIVELINESS`** **Quality of Service** (QoS) for **Data Distribution Service** (DDS) **Topics** where data is not sent sporadically; that is, it is sent with no fixed period.

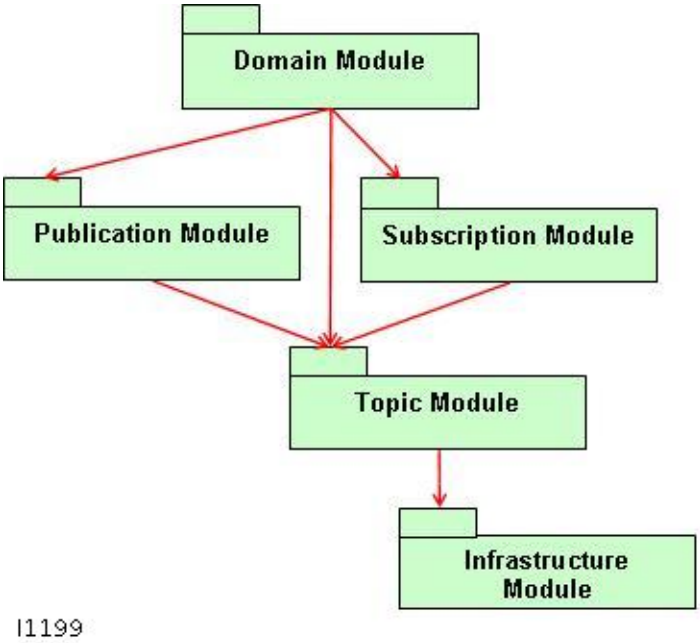
- BP1817: Use the **MANUAL_BY_TOPIC** setting of the **LIVELINESS** **Quality of Service** (QoS) for **Data Distribution Service** (DDS) **Topics** where the presence and health of the **DataWriter** is critical to the proper operation of the system.
- BP1818: Use the **HISTORY** **Quality of Service** (QoS) kind **KEEP_LAST** for **Data Distribution Service** (DDS) **Topics** that represent system state, in that new data-values replace the old values for each Keyed data-object.
- BP1819: Use the **HISTORY** **Quality of Service** (QoS) kind **KEEP_ALL** for **Data Distribution Service** (DDS) **Topics** that represent events or commands where all values written should be delivered to the readers (i.e., new values do not replace old values).
- BP1820: Use **TIME_BASED_FILTER** **Quality of Service** (QoS) to protect **DataReaders** that cannot handle all the traffic that could be written by the writers on that **Data Distribution Service** (DDS)**Topic** and just need periodic updates on the most current data-values.
- BP1821: Use the **Data Distribution Service** (DDS) **LIFESPAN** **Quality of Service** (QoS) to indicate that data is only valid for a finite time period and stale data is discarded after a certain expiration time elapses.
- BP1822: Use the **PARTITION** **Quality of Service** (QoS) to limit the scope of the data written/read on a **Data Distribution Service** (DDS) **Topic** to only the writer/readers that have a common partition.
- BP1823: Use the **Data Distribution Service** (DDS) **RESOURCES_LIMITS** **Quality of Service** (QoS) in platforms with limited memory or in **real-time systems** to properly configure the resources that will be utilized and avoid exhaustion of system resources at run-time.
- BP1824: Use the **USER_DATA** **Quality of Service** (QoS) to communicate metadata on the **DomainParticipant** that may be used to authenticate the application trying to join the Data **Distribution Service** (DDS) **Domain**.
- BP1826: Use the **USER_DATA** **Quality of Service** (QoS) on the **DataWriters** and **DataReaders** to communicate metadata that may provide application-specific information of the entity writing/reading data in a **Data Distribution Service** (DDS) **Domain**.
- BP1828: Use the **Data Distribution Service** (DDS) **OWNERSHIP** **Quality of Service** (QoS) kind set to **SHARED** when each unique data-object within a DDS **Topic** to which multiple **DataWriters** can write.
- BP1829: Use the **Data Distribution Service** (DDS) **OWNERSHIP** **Quality of Service** (QoS) kind set to **EXCLUSIVE** when multiple **DataWriters** cannot write each unique data-object within a DDS **Topic** simultaneously.

P1193: DDS Data-Centric Publish-Subscribe (DCPS)

The **Data-Centric Publish-Subscribe** (**DCPS**) interface is targeted toward the efficient delivery of the proper information to the proper recipients. It provides the application with a **data-centric** information model and is responsible for controlling the lower level layer of the DDS infrastructure targeted toward the efficient and reliable delivery of the information to its intended recipients. The DCPS architecture is comprised of five **modules**. The modules build upon each other in a hierarchical inheritance structure. The following table captures the purpose of each of the five modules.

Infrastructure Model	Defines the abstract classes and the interfaces that are refined by the other modules; also provides support for the two interaction styles (notification- and wait- based) within the middleware
Domain Module	Contains the **DomainParticipant** class that acts as an entry point of the Service and acts as a factory for many of the classes; the **`DomainParticipant`** also acts as a container for the other objects that make up the Service
Topic-Definition Module	Contains the **Topic**, **`ContentFilteredTopic`**, and **`MultiTopic`** classes, the **`TopicListener`** interface, and more generally, all that is needed by the application to define Topic objects and attach **QoS** policies to them
Publication Module	Contains the **Publisher** and **DataWriter** classes as well as the **`PublisherListener`** and **`DataWriterListener`** interfaces, and more generally, all that is needed on the publication side
Subscription Module	Contains the **Subscriber**, **DataReader**, **`ReadCondition`**,and **`QueryCondition`** classes, as well as the **`SubscriberListener`** and **`DataReaderListener`** interfaces, and more generally, all that is needed on the subscription side

The following is a UML Class diagram that represents the five modules and how they relate to each other.



Detailed Perspectives

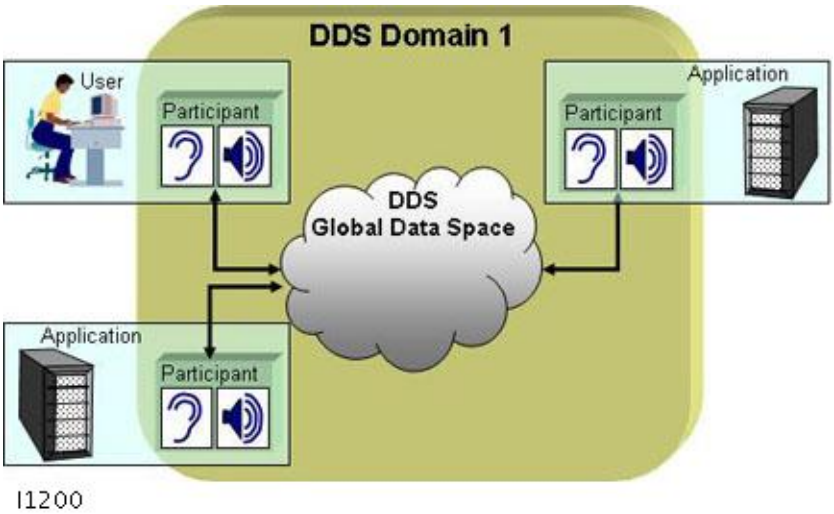
- DDS Domains - Global Data Spaces [P1194]

- Reading/Writing Objects within a DDS Domain [P1195]
- Messaging within a DDS Domain [P1196]

P1194: DDS Domains - Global Data Spaces

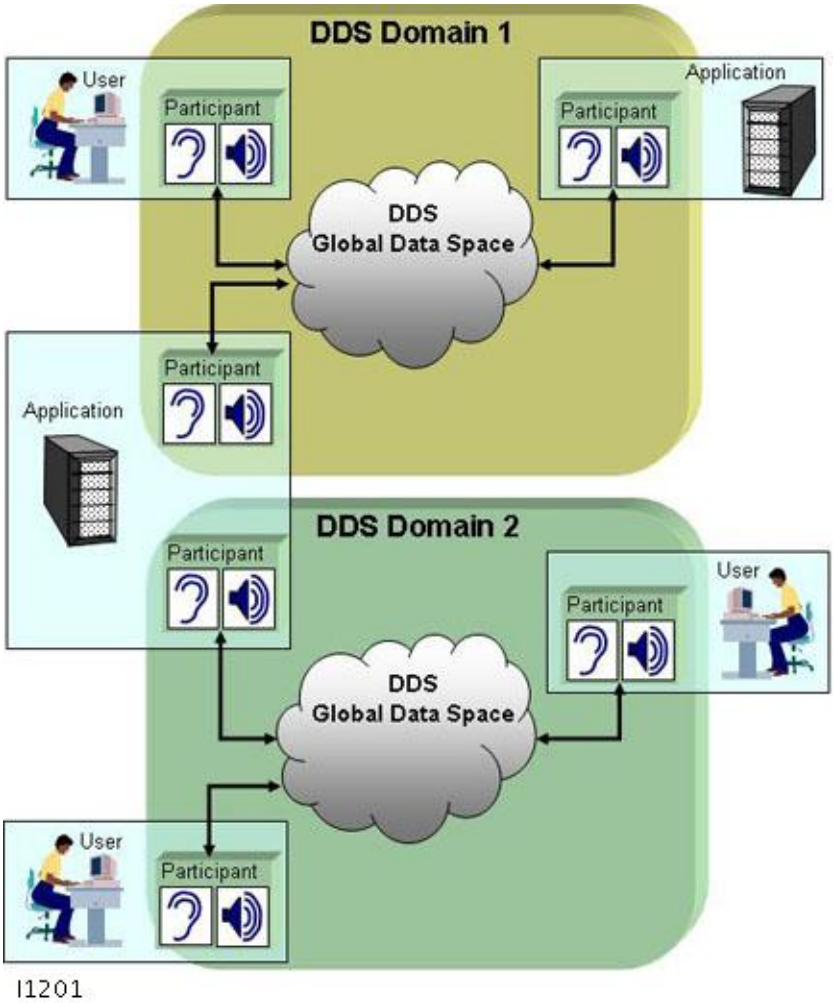
DDS allows application developers to create a collection of virtual shared **Global Data Spaces** where separate application processes can share data anonymously. Processes can access (read and/or write) data in the Global Data Space as well as exchange messages on the associated DDS **Domain**.

A DDS Global Data Space (called a DDS Domain) is identified by a **`domainId`** that represents an isolated Data Space. The Data Space exchanges no information or messages with other domains. The operating system maintains isolation between DDS Domains by using different port numbers. Each computer process (running on behalf of some user or application) must attach to the desired DDS Domain by creating a DDS **DomainParticipant**. Each **`DomainParticipant`** is owned by the creating process and is only accessible to it.



> ***Note:*** *The centralized image of a Global Data Space is just a convenient metaphor. In reality the DDS specification mandates that there should be no centralized implementation of the global data and data updates must flow directly from the writer to the readers.*

A distributed system may employ multiple DDS Domains (i.e., Global Data Spaces), each identified by a different **`domainId`**. A single application process may access multiple Global Data Spaces by creating multiple **`DomainParticipants`**, each associated with one of the Global Data Spaces.



Guidance

- G1770: Explicitly define **Data Distribution Service** (DDS) **Domains**.
- G1772: Assign a unique identifier for each **Data-Distribution Service** (DDS) **Domain**.

P1195: Reading/Writing Objects within a DDS Domain

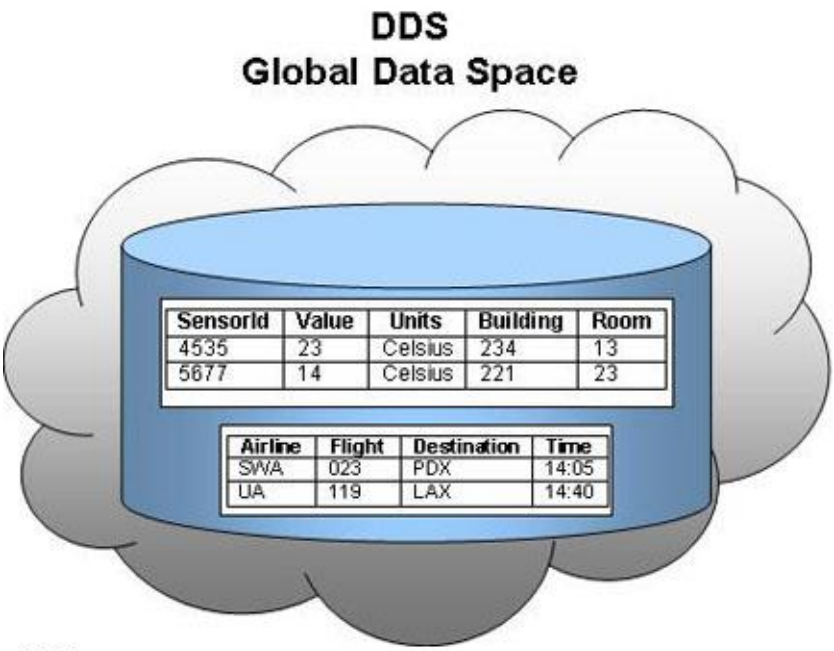
Address the Data Objects in the **Global Data Space** by means of a **Topic** (an application-chosen string that encodes a homogeneous collection of objects) and a ***Key*** (a set of fields inside the data object that uniquely identifies the object within the collection). A DDS Topic is an application-chosen string (such as **`Temperature`**) that has an associated schema or format representing the type of the data objects (for example the sensor ID, the value, the units, the location of the sensor, the time-stamp, etc.). The DDS Key is specific to each DDS Topic and uniquely identifies each Data Object within the Topic.

Pictorially one could think of each Topic in the Global Data Space representing a table of related data objects where each row represents the value of an individual data object the columns define the schema (data type of the object), and the key is the column(s) that defines the identity of each object. The table below depicts this concept for the hypothetical **`Temperature`** Topic.

SensorId (Key)	***Value*** : float	***Units*** : string	***Location*** : string	***Timestamp***
4535	23	Celsius	Building 234, Room 13	Tue Oct 31 15:47:42 PST 2006
5677	12	Celsius	Building 121, Furnace 23	Tue Oct 31 15:44:42 PST 2006

Another example is an Airport Information application that defines the Topic **`DepartingFlights`** with a schema consisting of fields containing the following information: Airline, flight number, destination airport, departure terminal, gate, scheduled departure time, expected departure time, and status. In this case the combination of fields Airline and Flight Number provides the Key that uniquely identifies each flight. Updates to the global data space will provide new estimated departure times, departing dates, etc. A display application may read this topic to show all the flights departing in the next three hours.

Airline (Key)	***Flight Number*** (Key)	***Destination***	***Departure Terminal***	***Departure Gate***	***Scheduled Departure***	***Expected Departure***	***Status***
SWA	023	PDX	A	12	10:30	14:05	Departed
UA	119	LAX	A	06	14:27	14:40	Boarding
AS	543	ANC	A	03	14:10	14:20	Boarding
KLM	006	AMS	A	14	14:35	14:35	Boarding
SQ	012	SIN	B	03	15:00	15:20	Go to Gate
JL	001	NRT	B	33	15:45	15:45	Go to Gate
LOT	007	WAW	B	02	16:30	16:30	Wait



I1202

Guidance

- G1810: Use **data models** to document the data contained within the **Data Distribution Service** (DDS) **Data-Centric Publish Subscribe** (DCPS).
- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1146: Include information in the **data model** necessary to generate a **data dictionary**.
- G1147: Use **domain analysis** to define the constraints on input data validation.
- G1148: **Normalize** data models.

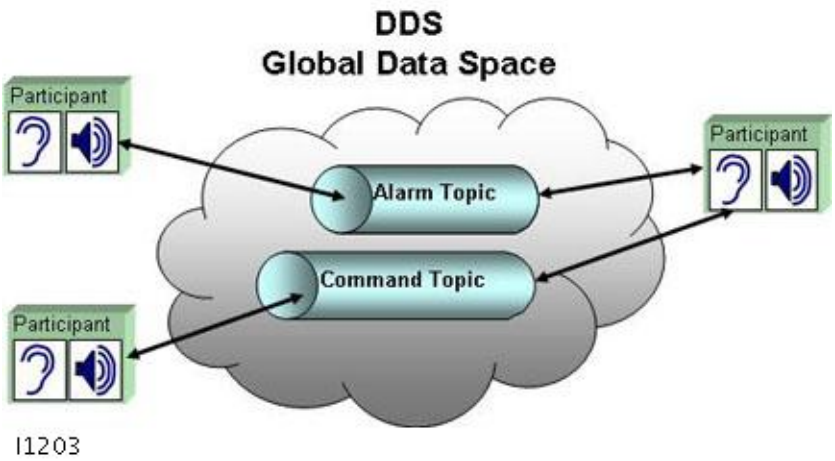
Best Practices

- BP1145: Use vendor-neutral **conceptual/logical models**.
- BP1254: For **command-and-control** systems, use the names defined in the Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM) for data exposed to the outside communities.
- BP1397: Identify and develop use cases or reuse existing use cases as appropriate as early in the data engineering process as possible to support **data model** development.

P1196: Messaging within a DDS Domain

A DDS **Topic** acts like a virtual message-queue or pipe when DDS is used for messaging. Writers send messages though the Topic and readers access messages using the same Topic.
Topics for DDS messages are bound to an application-defined schema in advance; for example, an ***Alarm*** message where the schema consists of source identifier, the kind of alarm, the location, a time-stamp, and the urgency level. **DomainParticipants** can publish and subscribe messages by specifying the Topic and the associated contents.

The Topics used for messaging also live within a DDS **Domain** (i.e., **Global Data Space**) identified by a unique **`DomainId`**. Similar to the data-object paradigm, the middleware keeps the messaging Topics separated within different DDS Domains by using different port numbers.



> ***Note:*** *The centralized image of a pipe is only a convenient concept. In reality, the DDS specification mandates that there should be no centralized implementation of a pipe in DDS. Messages must flow directly from the sender to the receivers.*

The distinction between reading/writing data and receiving/sending messages is essentially a property of the Topic. Some Topics represent data (if the identify certain fields as Keys) and others represent messages (if they do not contain specify Keys). In addition, use different **Quality of Service** settings to attain the proper semantics. For example, associate Topics representing data with a **`HISTORY`** QoS setting of **`KEEP_LAST`** whereas Messages typically use a **`HISTORY`** setting of **`KEEP_ALL`**.

> ***Note:*** *For more details on this subject please refer to the introductory material on DDS available at the OMG DDS Portal.*

Guidance

- G1796: Explicitly define **Data Distribution Service** (DDS) **Domain Topics**.
- G1798: Explicitly define all the **Data Distribution Service** (DDS) **Domain data types**.
- G1799: Explicitly associate data types to the **Data Distribution Service** (DDS) **Topics** within a DDS **Domain**
- G1800: Explicitly identify Keys within the **Data Distribution Service** (DDS) **data type** that uniquely identify an instance of a data object.
- G1801: Explicitly define a **Topic Quality of Service** (QoS) for each **Data Distribution Service** (DDS) Topic within a DDS **Domain**.

P1197: DDS Data Local Reconstruction Layer (DLRL)

The **Data Local Reconstruction Layer** (DLRL) is an optional part of the Data-Distribution Service (DDS) specification that provides a local ***object-cache*** abstraction built upon the core **DCPS** layer and requires application objects to comply with the DLRL object metamodel which includes collections and relationships.

> ***Note:*** *The DLRL, a recent addition to the DDS specification, is particularly rich; implementations using this upper-level profile of the specification are emerging.*

Application developers use the DLRL to do the following:

- Describe classes of objects with the associated methods, data fields and relations
- Attach data fields to Data-Centric Publish-Subscribe (DCPS) entities
- Use native language constructs to manipulate objects (i.e., create, read, update, delete) using native language constructs to seamlessly interact with the DCPS layer
- Manage objects and pointers to objects in a cache

Best Practices

- BP1832: Handle all **Data Distribution Service** (DDS) **Data Local Reconstruction Layer** (DLRL) Exceptions.
- BP1833: Use the **Data Distribution Service** (DDS) Object Model Profile for accessing message data as objects.

P1048: Messaging with MSMQ

Messaging in **.NET** uses Microsoft Message Queue (**MSMQ**). MSMQ is responsible for reliably delivering **messages** between applications inside and outside the enterprise. MSMQ ensures reliable delivery by placing messages that fail to reach their intended destination in a queue and then resending them once the destination is reachable.

I1067

MSMQ also supports transactions. It permits multiple operations on multiple queues, with all of the operations wrapped in a single transaction, thus ensuring that either all or none of the operations will take effect. Microsoft Distributed Transaction Coordinator (MSDTC) supports transactional access to MSMQ and other resources.

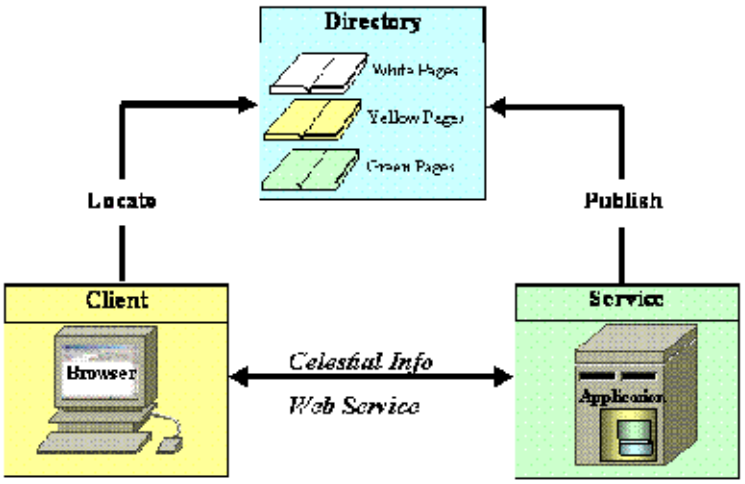
Best Practices

- BP1111: Mark all **Microsoft Message Queue** (**MSMQ**) messages as recoverable.
- BP1112: Specify all **Microsoft Message Queue** (**MSMQ**) queues as transactional if they support multiple-step processes.
- BP1227: Do not allow installation of **MSMQ**-dependent clients.
- BP1230: Do not use the **MSMQ** **`SupportLocalAccountsOrNT4`** feature.

P1078: Web Services

A **Web service** is an application that exists in a distributed environment, such as the **Internet**. A Web service accepts a request, performs its function based on the request, and returns a response. The request and the response can be part of the same operation, or they can occur separately in which case the consumer does not need to wait for a response. Both the request and the response usually take the form of **XML**, use a portable data-interchange format called **SOAP**, and are delivered over a **wire protocol**, such as **HTTP**.

A Web service can reside on top of existing legacy applications and expose services to the net. The Web services architecture illustrated below implements the **service-oriented architecture** pattern. For more information on design patterns, see "Web Service Patterns: Java Edition" by Paul B. Monday.



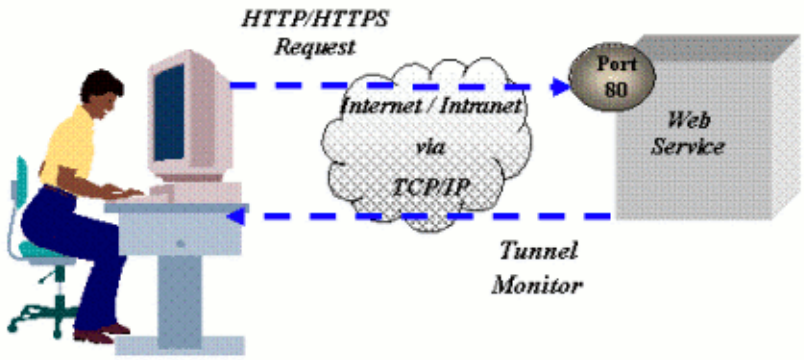
I1042

Web Service Models

Web services have traditionally been used to connect people to **services**. However, as the Web service infrastructure has matured, a new model has emerged, the service-to-service model.

Traditional Model

In a classic Web service, a request is usually made to a Web service using a **Web browser**. The request is submitted to the Web service using **HTTP** or **HTTPS** over the **Internet** or an **intranet**. The Web service processes the request and returns an **HTML** page that can be displayed in a Web browser.



I1043

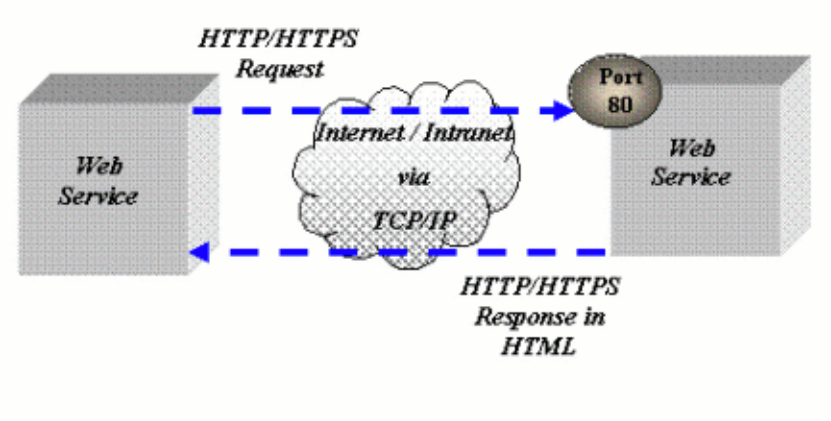
A classic Web service has the following characteristics:

- **Web pages** appear via a Web browser

- Connection is via **TCP/IP**
- Transport is HTTP/HTTPS
- Message format is HTML

Service-to-Service model

Application servers used to be responsible for providing machine-to-machine services. Now **Web servers** can handle similar work. The Web server can pass a request as an **XML** payload embedded in a TCP/IP and HTTP request, process the data, and respond. The response is typically in the form of an HTML Web page or an XML payload that a **client** application can use.



I1044

Machine-to-machine Web services have the following characteristics:

- Two independent applications
- Two independent **servers**
- Connection is via TCP/IP
- Transport is HTTP (port 80)
- Message format is XML payload in **SOAP** format

Key characteristics

Some key characteristics of Web services include the following:

- High-overhead interactions; may be too heavy for some applications
- **Loosely coupled** collaborators (e.g., client/server)
- Multiple layers of **parsing**, **marshalling**, and un-marshalling
- Non-standard content
- Standard interaction **protocol**
- No support for **services** such as **messaging** and security
- Infant technology
- No support for pass-by-reference

Guidance

- G1087: Validate all **Web Services Definition Language** (**WSDL**) files that describe **Web services**.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1090: Do not **hard-code** a **Web service's endpoint**.

P1068: SOAP

SOAP is an **XML** message-based **protocol**. SOAP is lighter weight and requires less programming than similar protocols such as **CORBA** and **Distributed Component Object Model** (**DCOM**). SOAP defines an extensible messaging framework independent of programming models and other implementation-specific semantics.

The **World Wide Web Consortium** (**W3C**) provides this description of SOAP:

"SOAP Version 1.2 (SOAP) is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics."

Two major design goals for SOAP are simplicity and extensibility. SOAP attempts to meet these goals by omitting distributed-system features from the messaging framework. Such features include but are not limited to reliability, security, correlation, routing, and Message Exchange Patterns (MEPs). While it is anticipated that many features will be defined, this specification provides specifics only for two MEPs. Other features are left to be defined as extensions by other specifications.

Key characteristics

SOAP is an XML message-based **wire protocol**.

SOAP is implemented by many language bindings.

SOAP is inherently stateless; consumers of SOAP services manage their own state.

SOAP relies on other standards to implement security directly.

Message styles

The **W3C WSDL** 1.1 Specification identifies two message styles: Document and RPC. The purpose of the styles determines how the content of the SOAP message body is formatted.

Document	The SOAP Body contains one or more child elements called parts. There are no SOAP formatting rules for what the SOAP Body contains; it contains whatever the sender and the receiver agree upon. ***Note:*** *There is a Wrapped form of this style that is required to interoperate with Microsoft* ***Web services*** *using Document style. There is no specification that defines this style.*
RPC	**RPC** implies that the SOAP Body contains an element with the name of the method or remote procedure being invoked. This element in turn contains an element for each parameter of that procedure.

Note: *Document style can be interpreted as either an* ***XML*** *string or as a W3C* ***Document Object Model*** *(****DOM****) Document Element. Microsoft has a technique called Wrapped that encapsulates the information being exchanged, regardless of the style.*

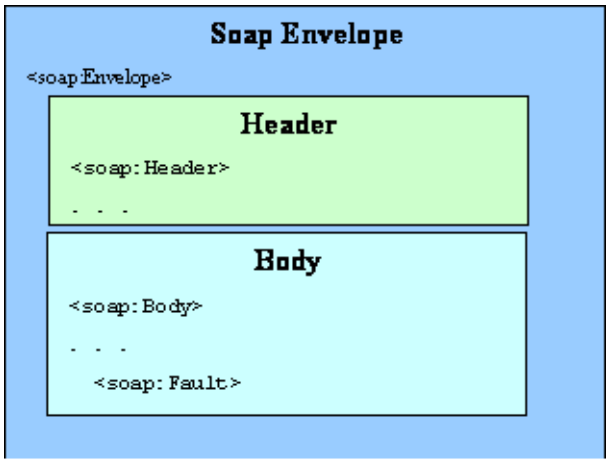
Serialization formats

For applications that use **serialization**/**deserialization** to abstract away the data wire format, there is one more choice to be made: the serialization format. The following table describes the two most popular serialization formats today.

SOAP encoding	SOAP encoding uses a set of rules to serialize the data transferred between the **client** and the **server**. The rules are defined in section 5 of the **WSDL** 1.1 Specification. These rules are also referred to as "section 5 encoding." The rules specify how to serialize objects, structures, arrays, and object graphs and directly use the predefined **XML Schema** data types. Generally, an application using SOAP encoding should use the **RPC mssage** style.
Literal	Data is serialized according to an independent external schema. There are no preset rules for serializing objects, structures, and graphics, etc., in the literal encoding style. The industry is overwhelmingly embracing XML Schemas.

Structure

A **SOAP** message comprises three parts: an envelope, an optional header, and a required body. The envelope encapsulates the other two elements. The optional header contains one or more header elements that contain meta-information about the method calls.



I1046

Envelope	The Envelope is the root of the SOAP request. At a minimum, it defines the SOAP namespace for SOAP 1.2. The envelope may define additional namespaces.
Header	The Header contains auxiliary information as SOAP blocks, such as authentication, routing information, or transaction identifier. The header is optional.
Body	The Body contains the main information in one or more SOAP blocks; for example, a SOAP block for **RPC** call. The body is mandatory and it must appear after the header.
Fault	The Fault is a special block that indicates a protocol-level error. If present, it must appear within a Body element.

Guidance

- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1093: Implement exception handlers for **SOAP**-based **Web services**.
- G1095: Use **W3C** fault codes for all **SOAP** faults.

P1081: Web Services Compliance

The **Web Services Interoperability Organization** (**WS-I**) is an open industry effort to promote **Web services** interoperability across platforms, applications, and programming languages.

The WS-I goal is to be a standards integrator to help Web services advance in a structured, coherent manner as standards evolve independently and in parallel. To support this, WS-I is developing a set of profiles that provide implementation guidelines for how to use related Web services specifications together for best interoperability.

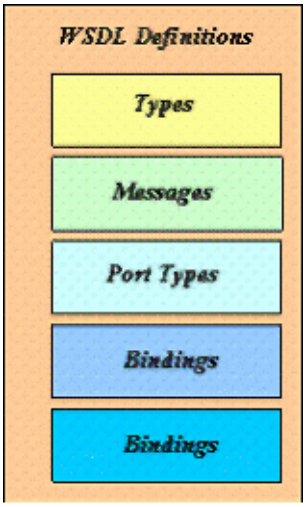
To date, WS-I has finalized the ***Basic Profile***, ***Attachments Profile*** and ***Simple SOAP Binding Profile***.

Guidance

- G1080: Adhere to the **Web Services Interoperability Organization** (**WS-I**) Basic Profile specification for **Web service** environments.
- G1082: Use the document-literal style for all data transferred using **SOAP** where the document uses the **World Wide Web Consortium** (**W3C**) **Document Object Model** (**DOM**).
- G1083: Do not pass **Web Services-Interoperability Organization** (**WS-I**) **Document Object Model** (**DOM**) documents as strings.

P1082: WSDL

The Web Services Description Language (**WSDL**) is an **XML**-based language that is used to describe a **Web service**. It describes the operations that are available from the Web service and it describes the data that flows between the **client** or consumer of the Web service and the producer of the Web service. In addition, it describes the **endpoint** The **URL** or location of the Web service on the internet of the Web service provider.



I1060

Guidance

- G1085: Establish a **registered namespace** in the **XML Gallery** in the **DoD Metadata Registry** for all DoD Programs.
- G1087: Validate all **Web Services Definition Language** (**WSDL**) files that describe **Web services**.

P1035: Insulation and Structure

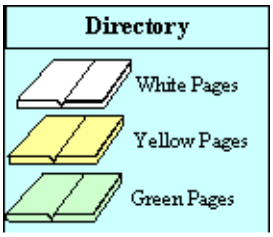
Insulating the user of **Web services** from the implementation of the services enhances the maintainability and portability of the overall system and aids in the migration to net-centricity. Application developers can use the facade or adapter design pattern for Web services to insulate applications from the implementation details of the service. Services can then change over time to match changing requirements and deployments. Legacy functionality can be similarly wrapped via a service. It is important to not directly expose vendor-specific functionality via the services interface to enable the ready reimplementation of the service if necessary.

Guidance

- G1087: Validate all **Web Services Definition Language** (**WSDL**) files that describe **Web services**.
- G1088: Use isolation **design patterns** to define system functionality that manipulates **Web services**.
- G1237: Do not **hard-code** the configuration data of a **Web service** vendor.
- G1090: Do not **hard-code** a **Web service's endpoint**.

P1075: Universal Description, Discovery, and Integration (UDDI)

The **Universal Description, Discovery, and Integration** (**UDDI**) standard is an industry initiative for a **Web services** registry. It enables businesses to access a universal pool of Web services. The UDDI registry contains yellow pages, white pages, and so-called "green pages," like a phone book.



I1062

White pages	List point of contact information, such as • Name • Address • Phone • Fax • email
Yellow pages	List services that are available from businesses, such as • Weather data • Software development • Project management
Green pages	List service properties, such as • Business processes • Service descriptions • Binding information • Categorization of services • XML version, type of encryption, and Document Type Definition (DTD)

UDDI is a platform-independent, open framework that allows automated consumers and suppliers to find each other, assess mutual compatibilities, negotiate terms, and build the relationship. It supports human interaction as well as machine-to-machine communication. People can use a UDDI browser to review services and find point-of-contact information (white pages), and business information (yellow pages).

Like the **Domain Name System** (**DNS**), the UDDI registry comprises a network of **servers** on the internet. It is a **SOAP**-based mechanism. The **API** specification focuses on the storage, organization, and architecture of the registry.

The UDDI project takes advantage of **World Wide Web Consortium** (**W3C**) and **Internet Engineering Task Force** (**IETF**) standards such as **eXtensible Markup Language** (**XML**) and **HTTP** and Domain Name System (DNS) **protocols**.

Guidance

- G1127: Use a **UDDI** specification that supports publishing discovery services.

- G1131: Use industry standard Universal Description, Discovery, and Integration (**UDDI**) **APIs** for all UDDI inquiries.

P1296: Service Definition Framework

A Service Definition Framework (SDF) provides a common frame of reference for service users, customers, developers, providers, and managers. Its structure and methodology enable full definition of the Service Access Points (SAPs) for a service. The purpose of the SDF is not to describe the internal workings of a service. Rather, it concentrates on defining the boundary conditions for accessing a service through its service access point. The SDF also includes specific technical parameters and engineering-level data that prospective service developers and providers can use to design and implement new enterprise service offerings.

Complete an SDF entry for each enterprise service. Subsequently, register each service in a service registry (e.g., the NCES Service Discovery service or the Air Force Service Management Tool). The SDF provides the basis for a design specification where potential implementers of a new service will find the information required to implement the service. The SDF should address the following information for each service:

- What the service does
- How the service works (from a black box perspective)
- Any required security mechanisms or restrictions
- Any pertinent performance or quality of service (Q0S) information
- Points of contact for the service:
  - Who is providing the service
  - Who is responsible for the daily operation of the service
  - Who is developing the service
- The specifics of how to bind to (access or use) the service.

Service Profiles

A service profile captures the black box architecture of a service. It would precede and guide one or more service implementations documented in association with the SDF. The use of a service profile becomes critical in the case of those enterprise services that have more than one implementation and implementer across the enterprise. The profile provides the guidance needed to ensure that multiple service implementations provide a common consumer interface and are interoperable.

Proposed SDF Lifecycle

The proposed SDF lifecycle is to assist service implementers in developing and maintaining an SDF entry during the lifecycle of an enterprise service. Scenarios include the following:

- Creating an SDF Entry
- Changing a Registered SDF Entry
- Deprecating a Registered SDF Entry
- Accessing a Registered SDF Entry

The proposed SDF Lifecycle is consistent with the DoD Acquisition Steps defined in the DoD 5000 series Directives and Instructions. The table below describes the proposed steps for the SDF lifecycle, along with associated business processes, the service owner and mandatory categories for each phase.

Lifecycle Element	***Description***	***Business Processes***	***Service Owner***	***Mandatory Categories by Phase***

Concept Development	Identify possible need for a new service and create justification for service	Examine mission threads and search for services to fulfill them. Identify capability gaps. These gaps become services within classification domains. Create high level business or mission capability statement. Perform initial cost analysis and Analysis of Alternatives. Define acquisition approach and organizations to execute following phase	Portfolio Manager	Service name, service description, schedule
Requirements and Architecture	Define service architecture and requirements	Identify specific organizations for each type of user, Define service requirements and semantics. Define service architecture to include interaction with other services and systems, basic service capabilities and service deployment approach. Perform Systems Program Office (SPO) level cost analysis.	Portfolio Manager to Acquirer	Semantic model, pedigree, information security marking, cpoints of contacts
Service Design	Create service "black box" interface specs for handoff to developers	Start configuration management: • finalize semantics • point to metadata repository • finalize classification details • determine service level agreements (SLAs) offered, finish WSDL	Acquirer	Operations, number of operations, security mechanisms, access criteria and restrictions, service level specification, network requirements, SAP
Service Build	Develop/purchase service	Development (generally follows contractor's best practices)	Acquirer	Consumer patterns, schedule Beta, operational reference
Service Testing	Assure service meets specifications and requirements	Acceptance test: • meets specifications • plays well with others • interoperability "seals of approval" from authoritative bodies	Acquirer to Operator/ Sustainer	Schedule: integration
Service Deployment	Install service instance(s)	Configuration management: • updating humans/summary from monitoring • measuring coarse-grained triggers for action (scaling)	Operator/ Sustainer	Schedule: deployment

Service Operation	Operate service; concludes with EOL announcement.	Configuration management: • updating humans/summary from monitoring • measuring coarse grained triggers for action (scaling)	Operator/ Sustainer	Schedule: operation
Service Deprecation	Service is still being operated but is to be replaced or retired; concludes with service EOL	Work with consumers to adopt new version of service, or replacement service(s) as appropriate	Operator/ Sustainer	Schedule: deprecation
Service Retired	Service is not operating; service definition information is still available for use/ reuse; concludes with purging of service definition information	Service migration and reuse	Sustainer	Schedule: retire

Notional SDF Concept of Operations

The Notional SDF Concept of Operations (CONOPS) outlines a theoretical concept for Service Discovery. The SDF concept focuses on why a service is needed and how it is used. The Notional SDF CONOPS addresses the following issues:

- Key Assumptions:
  - Location, composition, extensibility, syntax, failover, information assurance, alignment to COIs and applicable security classification level
  - Governance
  - Services are made available via an Enterprise Service Bus or via the Web services stack
  - The SDF will be used for defining services from many sources and multiple languages
- Creation of an SDF Entry
  - Two scenarios in which a service will require the creation of an SDF entry:
    - Capability already exists and will be "service enabled"
    - Capability does not exist
  - The SDF entry becomes part of the Key Interface Profile (KIP) for that service
- Services Lifecycle and SDF Development Process Flow
  - Establishment of a business case
    - Warfighter or COI has defined a need
    - Service requirements analysis and definition
    - Funding
    - Resources assigned
  - Design
  - Development
  - Test

    - Deploy
- SDF Implementation
    - SOA
        - Publishing
        - Discovery
        - Binding
    - Operations and maintenance
        - Change Management
        - Deprecation
        - Monitoring and maintenance

    Under SDF Implementation, NESI also advises that ConOps include Portfolio Management and Capability Planning. NESI will add these components in future versions.

SDF Considerations

- Describe all services using a standard Service Definition Framework (SDF).
    - Adhere to DoD Policy as a core definition for the SDF
    - Extensions can be made to core definition to suit specific needs
- May want to extend "Required" fields (from core SDF)
- Capture and track associated Lifecycle Phase
- The "Owner" of the service (and SDF) will change as the Lifecycle Phase changes; update the SDF at each Lifecycle phase.
- Begin capturing SDF data at the earliest possible Lifecycle Phase, preferably Concept Development.
    - Not all information will be available
    - Recommended to trace service capability back to operational needs, shortfalls and requirements
- Make SDF data accessible by storing contents either in an XML document in conformance with the XML Schema or in the form of a set of database tables with a front-end.
    - The XML Schema or database tables will contain all elements and attributes of the core (and extended) SDF
    - Common practices for database tables with a front-end include the following:
        - Group SDF data elements into logical categories and reflect such in the User Interface (UI) for ease of use; do not just provide one large input form
        - Reports are high value; being able to view SDF data via reports allows for relationships to be discovered and services to be managed (Portfolio Management, Capability Based Planning)
        - Role-based access for data editing is vital for information assurance and integrity; don't want Service Owner A to edit Service Owner B's SDF
        - Enforce security policies at the Data Level rather than at the application and/or UI level; provides stronger information assurance and accountability (audits); allows data entries and data fields to be customized to each user/role
- Capture SDF data from discrete choices (lists) rather than just "free text"; while free text can be searched via key word, it does not allow as much capability for data relationships and data mining.
- Make SDF data understandable and use terminology/labels relevant to the particular domain (enterprise).

- Designate minimally required data with respect to appropriate Lifecycle Phase needed for a complete understanding of the service at that phase.
    - Tie "Required" fields to lifecycle phases; some information may not be available at earlier phases, but would be required before eventually moving into a later phase.

SDF Template

The SDF Template provides a sample logical model to help the service implementer to understand the big picture for the Service Definition Framework. The logical SDF model, summarized in the following table, provides the primary service element categories and service element names. Each service element represents information that may or may not be relevant to the particular service being described. Some service elements may only be applicable during certain phases in the service lifecycle. Other service elements may not apply to specific technologies.

The attributes of a service that are necessary to effectively define and describe the service are identified within the SDF and organized into the following categories:

- Interface information
- Security information
- Service level information
- Implementation information
- Point of contract (POC) information
- Service Access Point (SAP) information

All categories, with the exception of the SAP, are abstract and allow defining the service so as to encourage semantic understanding of the service. The last category (SAP) is the concrete portion that is filled in after the service implementation and deployment. The SAP binds the abstract service specification to the concrete service interface as implemented by an actual process. Specific syntax, protocols and IP address required to use the functionality provided by the service are contained in the SAP.

In the table, the service elements have an associated cardinality for inclusion in the SDF. Cardinality is interpreted as follows:

- Cardinality = 1: Element is mandatory, one instance only
- Cardinality = 1..n: Element is mandatory, one to many ("n" = no upper limit, or upper limit is specified)
- Cardinality = 0..1: Element is optional, but limited to one instance if it is present
- Cardinality = 0..n: Element is optional, and there may be one instance or more if it is present.

Table 2 has an additional column, which is the recommended lifecycle phase where the given service element applies. A detailed specification of Service "Data" Elements will be included in a future release of NESI.

Service Category Element	***Service Element***	***Cardinality***	***Service Development Lifecycle Phase***
Interface information	ServiceName	1	Concept Development
	Service Description	1	Concept Development
	Semantic Model	0..1	Requirements & Architecture
	NumberOfDataTypes	1	Service Design
	DataTypes	0..n	Service Design
	NumberOfOperations	1	Service Design

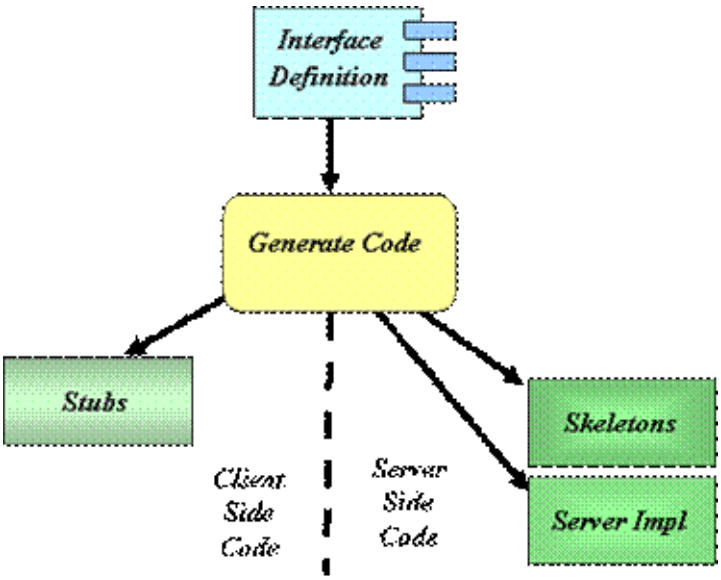
	Operations	1..n	Service Design
	ServicePedigree	1	Requirements & Architecture
Security information	SecurityMechanisms	1	Service Design
	AccessCriteriaAndRestrictions	1	Service Design
	InformationSecurityMarking	1	Requirements & Architecture
Service level information	NumberOfServiceLevels	1	Service Design
	ServiceLevelSpecifications	0..n	Service Design
	NetworkRequirements	0..1	Service Design
Implementation information	ConsumerPatterns	0..1	Service Build
	NumberOfScheduleDates	1	Concept Development
	Schedule	1..n	Concept Development
	NumberOfOperationalReferences	1	Service Build
	OperationalReference	0..n	Service Build
	VersioningApproach	0..n	Service Design
POC information	NumberOfContacts	1	Requirements & Architecture
	Contacts	1..n	Requirements & Architecture
SAP information	NumberOfSAPs	1	Service Design
	ServiceAccessPoint	0..n	Service Design

P1011: CORBA

CORBA is the acronym for **Common Object Request Broker Architecture**. It is the **Object Management Group** (**OMG**) open, vendor-independent architecture and infrastructure that computer applications use to work together over networks. Using the Internet InterORB **Protocol** (**IIOP**), a CORBA-based program from any vendor, on almost any computer, operating system, programming language, or network, can interoperate with a CORBA-based program from the same or another vendor on almost any other computer, operating system, programming language, or network.

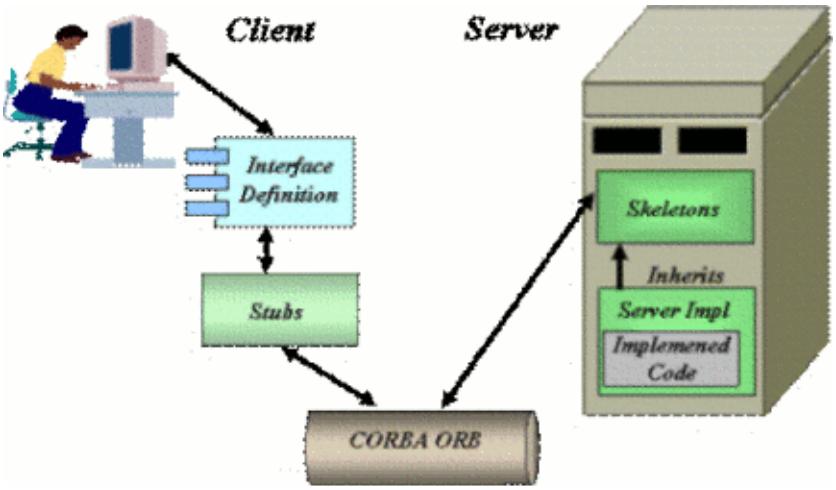
In general, the code that needs to be created to access an object remotely using CORBA can be implemented using well established and well understood design patterns. Consequently, it is not difficult to write but it is tedious and subject to human error during the writing process because much of it is of a cut-and-paste nature. Therefore, most **Object Request Broker** (**ORB**) vendors have developed code generators that can auto-generate the required infrastructure code given the definition of the interface between a **client** and a **server**. The use of these auto-generators is strongly encouraged.

The following diagram illustrates auto-generation of the infrastructure code from an interface defined using the CORBA **Interface Definition Language** (**IDL**).



I1069

This diagram illustrates how the generated code is used within the **CORBA** infrastructure.



I1071

Key features

Some of the key features of interest in the CORBA specifications follow:

- Internet InterORB Protocol (IIOP)
- Dynamic Invocation Interface (DII)
- Dynamic Skeleton Interface (DSI)
- Interface Repository (IFR)
- Objects by Value (OBV)
- CORBA Component Model (CCM)
- Portable Object Adapter (POA)
- General InterORB Protocol (GIOP)
- Java to Interface Definition Language (IDL) mapping

Guidance

- G1118: Localize **CORBA** vendor-specific source code into separate **modules**.
- G1202: Use the **CORBA Portable Object Adapter** (**POA**) instead of the **Basic Object Adapter** (**BOA**).
- G1119: Isolate user-modifiable configuration parameters from the **CORBA** application source code.
- G1204: Create configuration services to provide distributed user control of the appropriate configuration parameters.
- G1205: Use non-source code persistence to store all user-modifiable **CORBA** service configuration parameters.
- G1121: Do not modify **CORBA** Interface Definition Language (**IDL**) compiler auto-generated stubs and skeletons.
- G1123: Use the Fat Operation Technique in **IDL** operator invocation.
- G1203: Localize frequently used **CORBA**-specific code in **modules** that multiple applications can use.

Best Practices

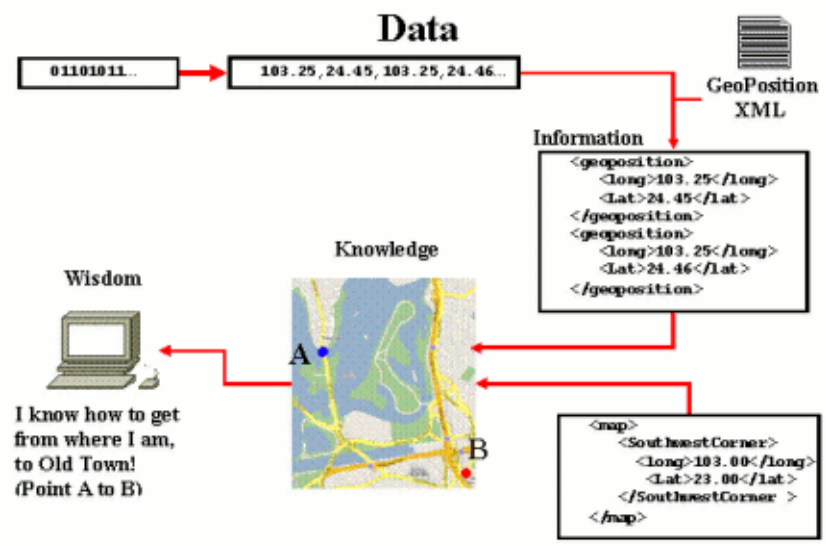
- BP1231: Use **`CORBA::String_var`** in **IDL** to pass string types in C++.
- BP1232: Do not pass or return a zero or null pointer; instead, pass an empty string.
- BP1233: Do not assign **`CORBA::String_var`** type to **`INOUT`** method parameters.
- BP1234: Assign string values to **`OUT`** , **`INOUT`** , or **`RETURN`** parameters using operations to allocate or duplicate values rather than creating and deleting values.
- BP1235: Assign string values to returned-as-attribute values using operations to allocate or duplicate values rather than creating and deleting values.

P1012: Data

There are several common definitions of data; the NESI Glossary definition includes the following points:

- **Data** is unprocessed **information**.
- Data is information without context.

But both of these definitions rely on the term "information" which can be a circular definition back to data. To clarify this, the following model helps create definitions of **Information**, **Knowledge** and **Wisdom**. Data flows into the **system** as a set of zeros and ones. The system transforms this initial data into other data that is more understandable from a human perspective (i.e., a list of double precision, floating point numbers). If the numbers are placed into a context such as it is a geographic position, then the data starts to become Information. As information is combined together, the result is referred to as Knowledge (i.e., the knowledge of where one is). When the knowledge can support making decisions, the results are Wisdom (i.e., how to get from point A to point B).



I1112

Within NESI, the term Data covers the entire data spectrum (i.e., Information, Knowledge and Wisdom) with a focus is on the transfer of data between components. There have been several major efforts within the **DoD** that have addressed the need to understand, control and document the flow of data between components. NESI is not in competition with these efforts nor is it intended to render these efforts obsolete. NESI provides detailed guidance intended to verify that the concepts and **tenets** of these efforts are met.

Generic data guidance statements include guidelines relative to basic functions associated with the definition of data and the most general categories of data types. Examples of the most basic data functions include **data modeling** and **domain analysis**. The most general categories of data types include **relational database** data and **XML**.

Data Exposure defines the steps necessary to set up the **metadata** infrastructure associated with a net-centric data strategy. This infrastructure permits the exposure (i.e., visibility) of net-centric data to the user community. This infrastructure will be set up once but maintained to include the following:

- Registry where the metadata will reside
- Repository where the data will reside
- Rules applicable to the tagging of data

Tagging and metadata rules follow from **Data Categorization**. Generic Data Categorization includes data types that adhere to **XML Schema** rules. Specialty Data Categories, such as **Electronic Data Interchange** (**EDI**) and **Binary XML** include data types that do not fit in the current XML paradigm but for which special XML extensions may be developed.

Data Publishing defines the steps necessary to make data available within the net-centric data strategy infrastructure. It requires the project to have a **Community of Interest** (**COI**), a model of the data associated with the project and an

ontology which taken together can be used as a basis for structural metadata. Based on the Data Categorization rules promulgated in the data exposure section appropriate tags are determined and applied to the data.

There are many ways to persist data to include storing data on a **file system** or in a database (e.g. **hierarchical databases**, **object-oriented databases**, **native XML databases**, and **relational databases**).

Detailed Perspectives

- XML [P1083]
- Metadata Registry [P1050]
- Data Modeling [P1003]
- Metadata [P1049]
- Relational Database Management Systems [P1063]

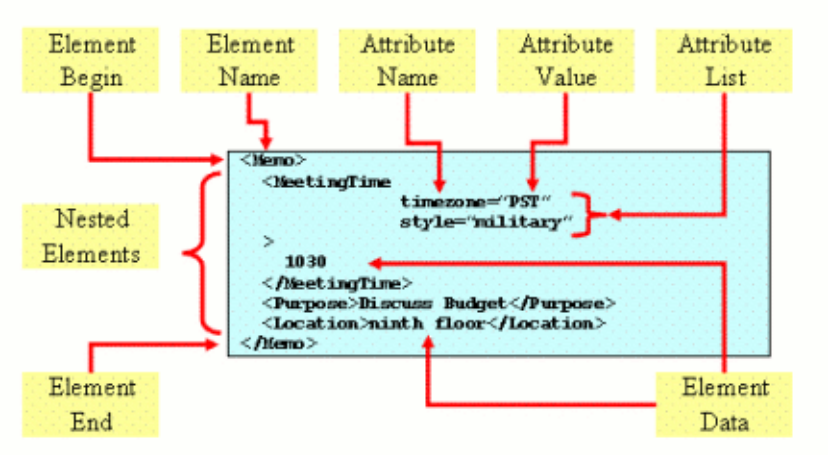
P1083: XML

The **Extensible Markup Language** (**XML**) is a **World Wide Web Consortium** (**W3C**) initiative that allows encoding **data** and information with meaningful structure and **semantics** into a document that computers and humans can read easily. XML is ideal for information exchange and is easily extended to include other data types. The ubiquitous nature of XML within existing and proposed DoD projects has spawned a lot of activity to capture guidelines and requirements that facilitate net-centricity and interoperability. Many of these activities have not been finalized and are "emerging" from a NESI viewpoint. This NESI Perspective leverages the work done by Roger Costello and colleagues at xFront.com. It is by no means complete, but it does provide a starting point for additional DoD XML work.
There are two key measures of XML instance document correctness: being **well-formed** and **valid**. Those concepts and others are introduced in the following perspectives:

- XML Syntax [P1095]
- XML Semantics [P1096]
- XML Processing [P1105]

P1095: XML Syntax

The syntax of an **XML document** is a hierarchical collection of **XML elements** that identify the name of the **data** within the XML document and the value associated with the element. Elements can have **attributes** and be nested within other elements. The following is a simplistic XML document displayed in **ASCII** with the major syntactical **components** labeled.



I1173

Guidance

- G1724: Develop XML documents to be well formed.

Best Practices

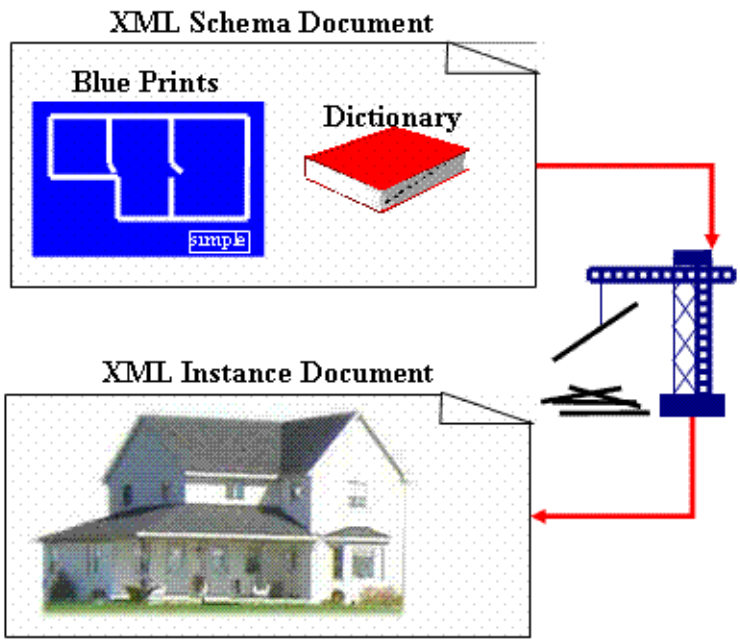
- BP1258: Explicitly define the encoding style of all data transferred via **XML**.
- BP1752: Place dynamic **XML element** data within an XML CDATA section.

Examples

An example of an XML instance document is the following weather information XML. It can be thought of as a complex data structure that contains a weather station's data.

P1096: XML Semantics

The semantics of an **XML document** are limited to the structural composition of data, the relationships of the structures to each other, and the rules governing data content. A full semantic interpretation of the **XML** content must be left to humans or tools that humans have written that connote some meaning to the data. For example, the semantics captured by XML might define a weather station that is comprised of air temperature, soil temperature, anemometer and hygrometer and the values and units associated with these values. XML does not capture what this data means semantically to a pilot or soldier.



I1174

The semantics of any XML instance document are captured in another XML document called the schema which is also defined using XML. Therefore, the semantics discussion is divided into two sub-perspectives:

- XML Schema Documents [P1097]
- XML Instance Documents [P1104]

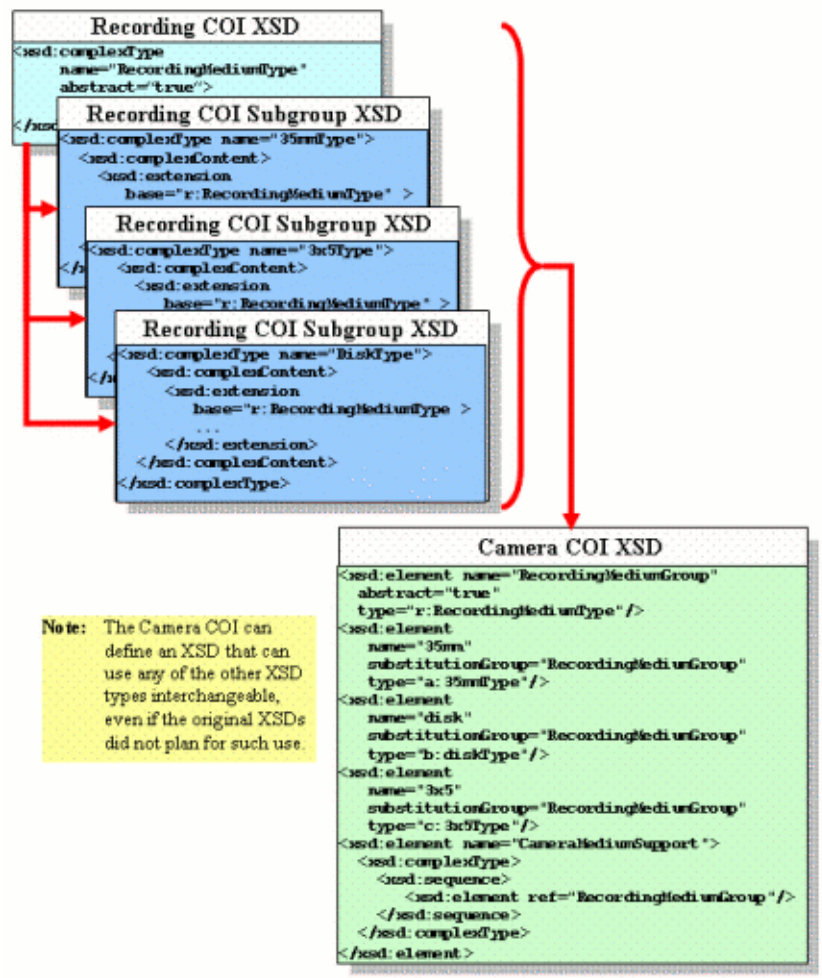
P1097: XML Schema Documents

An **XML Schema** is a **W3C** specification for defining the **semantics** and structure of **XML documents**. For a discussion of the grammar that governs **XML** see the XML Syntax [P1095] perspective. The semantics are limited to the structural composition of data, the relationships of the structures to each other, and the rules governing data content. The discussions of the schema documents are broken down into schema subject areas:

- Defining XML Schemas [P1098]
- XML Schema Files [P1099]
- Using XML Namespaces [P1100]
- Defining XML Types [P1101]
- Using XML Substitution Groups [P1102]
- Versioning XML Schemas [P1103]

P1102: Using XML Substitution Groups

Substitution groups allow using elements defined in externally defined and controlled schemas as interchangeable elements in new schemas. The members of the substitution group do not have to be derived from the same type. This allows any of the element members' substitution group elements to participate as a member of a more abstract concept. For example, in the following **XML**, **`RecordingMedium`** is the name of the substitution group. The members of the group are the **`RecordingMedium`** element itself and **`35mm`**, **`disk`** and **`3x5`**. Anywhere that **`RecordingMedium`** is used as a reference, **`35mm`**, **`disk`** and **`3x5`** can also be used. For a complete example study the following diagram that defines a **`CameraMediumSupport`** element that has a single sequence comprised of the **`RecordingMediumGroup`** substitution group.



I1175

Guidance

- G1731: Only reference **XML elements** defined by a Type in substitution groups.
- G1744: Only reference abstract **XML elements** in substitution groups.
- G1745: Append the suffix Group to substitution group **XML element** names.

P1101: Defining XML Types

The **W3C** defined datatype as follows:

"A datatype is a 3-tuple, consisting of a) a set of distinct values, called its value space, b) a set of lexical representations, called its lexical space, and c) a set of facets that characterize properties of the value space, individual values or lexical items."
[See W3C "XML Schema Part 2: Datatypes Second Edition," Section 2.1, http://www.w3.org/TR/xmlschema-2/#typesystem]
There are two kinds of datatypes definable within XML: Primitive and Derived. Primitive datatypes are not defined in terms of other datatypes while Derived datatypes are defined in terms of other datatypes. All datatypes can be further classified as Built-in and User-derived. Built-in datatypes are those which have been defined by the W3C in XML Schema Part 2: Datatypes Second Edition. User-derived datatypes are those defined by individual schema designers.

The guidance included in this perspective is for primitive and derived datatypes designed by individual schema designers.

Guidance

- G1727: Provide names for XML type definitions.
- G1728: Define types for all **XML elements**.
- G1729: Annotate XML type definitions.
- G1740: Append the suffix Type to XML type names.

Best Practices

- BP1732: Follow the Upper Camel Case (UCC) naming convention for XML Type names.

P1099: XML Schema Files

Schema definitions are usually captured in files. The following guidance applies to those files which actually contain the schema definitions.

Guidance

- G1735: Use the **`.xsd`** file extension for files that contain XML Schema definitions.
- G1736: Separate document schema definition and document instance into separate documents.

Examples

```
<?xml version="1.0"?>
<xsd:schema xmlns: xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.camera.org"
            xmlns: nikon="http://www.nikon.com"
            xmlns: olympus="http://www.olympus.com"
            xmlns: pentax="http://www.pentax.com"
            elementFormDefault="unqualified">
 <xsd:import namespace="http://www.nikon.com"/>
 <xsd:import namespace="http://www.olympus.com"/>
 <xsd:import namespace="http://www.pentax.com"/>
 <xsd:element name="Camera">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="body"
                     type="nikon:BodyType"/>
       <xsd:element name="lens"
                     type="olympus:LensType"/>
       <xsd:element name="ManualAdapter"
                     type="pentax:manual_adapter_type"/>
     </xsd:sequence>
   </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

P1100: Using XML Namespaces

A **namespace** defines the scope for schema components and de-conflicts the use of schema components. Qualifying prefixes simplify the use of namespaces in names by appending a qualifier onto the beginning of the name that is mapped to a particular schema. Namespaces can become quite confusing if they are not used consistently.

Guidance

- G1737: Define a target namespace in schemas.
- G1738: Define a qualified namespace for the target namespace.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1085: Establish a **registered namespace** in the **XML Gallery** in the **DoD Metadata Registry** for all DoD Programs.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.

Best Practices

- BP1739: Use the xsd qualifying prefix for XML Schema namespace.
- BP1741: Do not provide a schema location in import statements in schemas.
- BP1742: Use the xsi qualifying prefix for XML Schema instance namespace uses.

P1098: Defining XML Schemas

While it is possible to use **Document Type Definitions** (**DTD**) to convey much of the same information as the **XML Schema Definition** (**XSD**), XSDs have several distinct advantages which are very useful in terms of interoperability. **XML Schemas** have richer support for defining and using types than DTDs which capture domain information such as allowable ranges and units. For example, XSDs can define an elevation type with values limited to meters in the range of 0 to 12,000.

Guidance

- G1725: Develop XML documents to be **valid** XML.
- G1726: Define XML Schemas using **XML Schema Definition** (XSD).
- G1730: Follow an XML coding standard for defining schemas.
- G1045: Define **XML** format information separately in **XSL**.

Best Practices

- BP1732: Follow the Upper Camel Case (UCC) naming convention for XML Type names.
- BP1733: Follow the Upper Camel Case (UCC) naming convention for **XML element** names.
- BP1734: Follow the Lower Camel Case (LCC) naming convention for **XML attributes**.

P1103: Versioning XML Schemas

XML Schemas capture the **semantics** of the **data** that the schemas define. As the understanding of the data and its interrelationships evolves, the need to redefine the semantics captured by the schema is inevitable. This evolution can have a wide ranging ripple effect throughout a large widely distributed system or family of systems. Therefore, the uniform managing of schema versions is essential.

Guidance

- G1753: Declare the XML schema version with an **XML attribute** in the root **XML element** of the schema definition.
- G1754: Give each new XML schema version a unique URL.
- G1727: Provide names for XML type definitions.
- G1004: Make public **interfaces** backward-compatible within the constraints of a published **deprecation** policy.
- G1019: Deprecate public interfaces in accordance with a published deprecation policy.

P1104: XML Instance Documents

An **XML instance document** is an **XML document** which is defined by an **XML Schema** but is populated with the actual data whereas the schema is the definition of the structure and semantics of data (**metadata**).

Guidance

- G1725: Develop XML documents to be **valid** XML.
- G1736: Separate document schema definition and document instance into separate documents.

Best Practices

- BP1742: Use the xsi qualifying prefix for XML Schema instance namespace uses.
- BP1743: Use .xml as the file extension for files that contain XML Instance Documents.

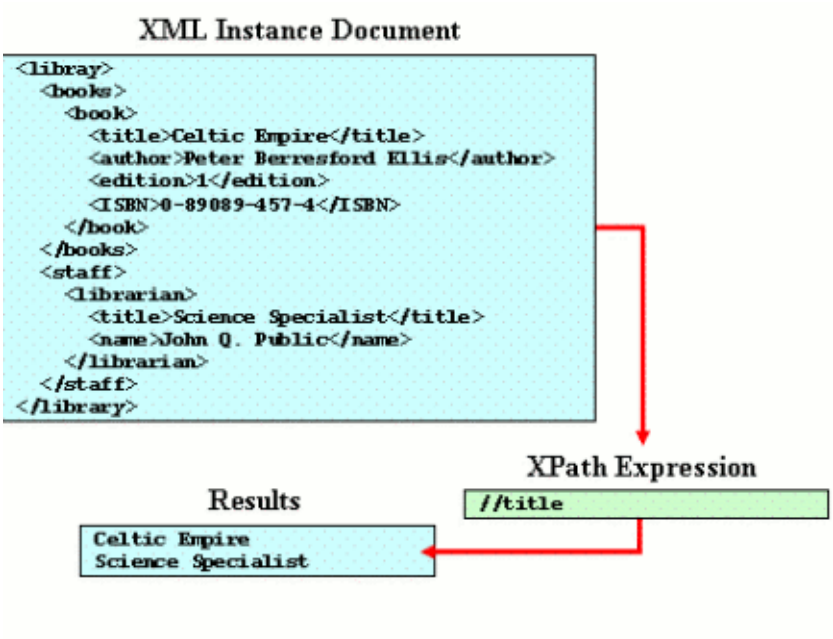
P1105: XML Processing

One of the primary benefits of using **XML** is that it can be read by humans or processed by software. The following perspectives pertain to XML processing:

- XSLT [P1106]
- XPath [P1107]
- Parsing XML [P1109]
- XML Validation [P1110]

P1107: XPath

A **valid XML Document** is a representation of a **Document Object Model** (**DOM**) tree structure. Each of the XML elements is considered a node with the tree. **XML Path Language** (**XPath**) is a succinct and elegant way of addressing the individual nodes (i.e., elements) within the tree (i.e., document) or to perform basic computations on the Element Data within the document. The following is a very simplistic example of how an XML Document and XPath work together. The XML instance document contains the data and the XPath provides the instructions on how to traverse the document.



I1172

For a more detailed description of XPath, see the following W3C location: http://www.w3.org/TR/xpath; there also is an XPath tutorial at http://www.w3schools.com/xpath/default.asp.

Guidance

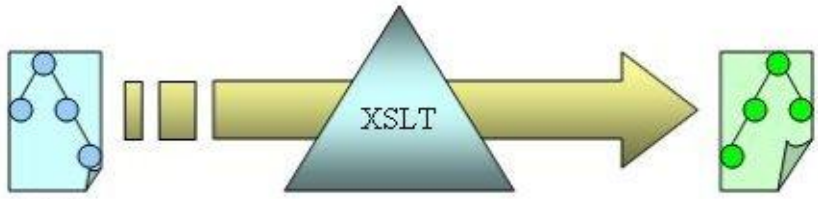
- G1756: Isolate XPath expression statements into the configuration data.

Best Practices

- BP1757: Do not ignore namespace prefixes in XPath expressions.
- BP1758: Make names in descendant expressions unique within an XML document.

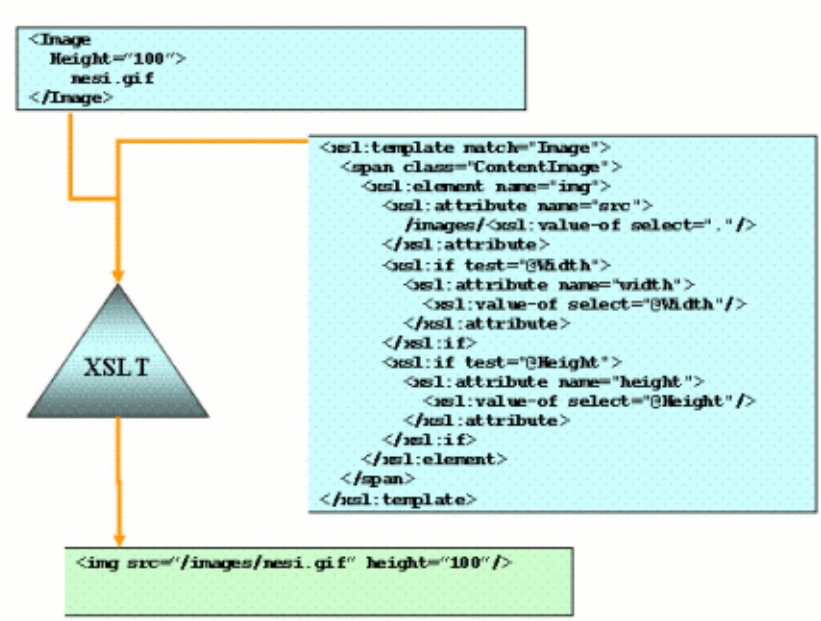
P1106: XSLT

XSL Transformations (**XSLT**) allow **XML** data transformation using the functional **eXtensible Stylesheet Language** (**XSL**).



I1176

XSL is dependent on **XML Path Language** (**XPath**) to address nodes within the input document. For XPath guidance and best practices see the XPath [P1107] perspective. The following example produces **HTML** image tag from an image **XML element** with optional height and width attributes.



I1177

Templates

Use templates to transform particular sections of an XML document tree. XSLT requires at least one template which matches to an absolute path of an element (e.g., **/**). Inside of a template, match other templates by using **`xsl:apply-templates`**. Passing an XPath query to the select parameter of **`xsl:apply-templates`** constructs a list of nodes by which templates are compared and executed.

XSLT 2.0

XSLT 2.0 improves on XSLT 1.0 and adds functionality that was previously only achieved through proprietary language extensions.

Some of the more significant improvements include the following:

- Backwards-compatibility
- Improved XPath functions
- Regular expressions

- Schema validation to temporal and result trees
- Multiple outputs
- Aggregation
- Strong data typing

Guidance

- G1746: Develop XSLT **style sheets** that are XSLT version agnostic.
- G1751: Document all XSLT code.
- G1755: Use accepted file extensions for all files that contain XSL code.

Best Practices

- BP1747: Use the xsl qualifying prefix for XSLT namespace.
- BP1748: Separate static content from transformational logic in XSLTs.
- BP1749: Use xsl:include for including XSL transforms.
- BP1750: Use xsl:import for reusing XSL code.

P1109: Parsing XML

One advantage of **XML** is that a variety of standard **parsers** are available to parse documents. Another advantage is that the consumer of the XML document is free to choose the type of parser to use.

A couple of common types of XML parsers include the **Document Object Model** (**DOM**) and Simple API for XML (SAX) parsers. The DOM parser uses a tree-based approach, while the SAX parsers use an event-based approach. Both approaches have advantages and disadvantages depending the application.

In addition to the various types of XML parsers, there are multiple implementations of each types of parser. This provides the developer great flexibility in choosing an XML parser implementation. To take advantage of this flexibility, the developer must take care when developing software to allow for changing the XML parser throughout the life-cycle of the software. One way to do this is to provide a wrapper or adapter class that isolates the XML parser implementation allowing for changes to the XML parser during development or deployment.

Best Practices

- BP1769: Provide wrapper or adapter classes to isolate XML parser implementations.

P1110: XML Validation

One advantage of **XML** is that it allows for validation of **XML instance documents**. Validation can occur at the producer and/or consumer or anywhere in-between.

Guidance

- G1725: Develop XML documents to be **valid** XML.

Best Practices

- BP1265: Validate **XML** documents during document generation.

P1050: Metadata Registry

A Metadata Registry is a central repository for storing and maintaining **metadata** definitions. A metadata registry typically has the following characteristics:

- It is a protected area where only approved individuals may make changes
- It stores **data elements** that include both semantics and representations
- The semantic areas of a metadata registry contain the meaning of a **Data Element** with precise definitions
- The representational areas define how the data is represented in a specific format such as within a database or a structure file format such as **XML**

Metadata registries often are stored in an international format called **ISO-11179**.

A metadata registry is frequently set up and administered by an organization's **Data architect** or data modeling team.

The **DoD Metadata Registry** provides a common source of data information required to promote interoperability in the Net-Centric Data Environment.

In the Net-Centric Data Strategy, data sources are called **Data Assets** which are divided into two generic areas:

The **data** area includes the following:

- **XML** stored in repositories (files)
- **Database data**
- Data services
- Data streams (real time)
- Sensor data
- **Message** data (includes **EDI**)

The **metadata** area includes the following:

- Metadata stored in registries
  - **UDDI**
  - **ebXML**
  - DoD Metadata Registry
  - Other **ISO/IEC 11179 Registries**
  - Discovery metadata stored in Catalogs
- DoD Discovery Metadata Standard (**DDMS**)
- Interface Metadata (**WSDL**)
- Structural Metadata (**XSD**)

Data comes in many forms. It can be simple or complex; structured or unstructured in nature.

Simple Structured Data has an uncomplicated **data structure** . All requisite metadata is provided and simple data types only are used (e.g., integers, long integers, strings, and simple lists).

Simple Unstructured Data has uncomplicated data structure but not all requisite metadata is provided.

Complex Structured Data has well-defined metadata. It includes data represented in **XML documents** with deeply hierarchical and recursive structures. Complex data can be represented in a complex data structure or can be mapped

into a relational or flat structure with additional metadata provided to represent the complex relationships. Although complex structured data is generically a property of object oriented databases, the Complex Data Structures can be filled from any source.

- Data
  - XML files
  - defined by **XML Schemas** (**XSDs**)
    - **Interface**
- Metadata stored in DoD Repository
  - XML Schemas (XSDs)
  - Discovery metadata
    - **WSDL**
    - **UDDI**
  - Web Service Source Code
  - XSDs include element validation and descriptions
  - XSDs may import other XSDs
  - XSDs are validated
  - **Complex Structured Data** follows all of the **XML** rules.

> ***Note:*** *The source of this data can be any.*

Complex Semi-Structured Data has partial metadata. It includes data defined in **COBOL** copybooks and Electronic Data Interchange standards **ANSI X.12** and Health Level 7 (HL7). Semi-structured data can be as complex or more so as any Complex Structured data. It can map into or be XML. It may also be missing some Metadata or an XSD.

Complex Unstructured Data has little or no metadata. It includes data in binary files, spreadsheets, documents, and print streams.

Guidance

- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1383: Use a **registered namespace** in the XML Gallery in the **DoD Metadata Registry**.
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1385: Identify **XML Information Resources** for registration in the XML Gallery of the **DoD Metadata Registry**.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1387: Identify **data elements** created during Program development for registering in the **Data Element Gallery** of the **DoD Metadata Registry**.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1389: Publish database tables which are of common interest by registering them in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1125: Use the **Department of Defense Metadata Specification** (**DDMS**) for standardized tags and taxonomies.

Best Practices

- BP1392: Register services in accordance with a documented service registration plan.

P1003: Data Modeling

Modeling is an essential step in understanding the data that will comprise a system. Before implementing a system, it is important to understand the basic **data elements** and the relationships of the elements. The end products of **data modeling** can be **XML schemas**, **RDBMS** schema definitions or the data portion of objects.

Rather than conducting data modeling efforts in isolation, seek out and identify relevant **communities of interest** (**COIs**). Doing so will provide for more effective data models that build upon lessons learned, provide lessons learned to the greater community, reduce costs through reuse, and enhance interoperability through the use of common semantics across the community. One way to do this is to base new data models on the terminology published by relevant COIs listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**. Another is to look for relevant COIs outside of the DoD. Examples of common high level COI data models follow.

Universal Core (UCore)

UCore is a federal information sharing initiative that supports the *National Strategy for Information Sharing* (available at http://georgewbush-whitehouse.archives.gov/nsc/infosharing/index.html) and associated Departmental and Agency strategies. UCore enables information sharing by defining an implementable specification (XML Schema) containing agreed upon representations for the most commonly shared and universally understood concepts of who, what, when, and where.

UCore is designed to be simple to understand, explain, and implement. It is small, containing a minimal set of objects with broad applicability across a wide range of domains. UCore is built on an extensible framework that permits users to build more detailed exchanges tailored to their mission or business requirements. UCore is based on and leverages existing commercial and governmental standards. The UCore validation processes and tools provide a means to achieve consistently definable levels of interoperability, promoting machine understanding between both anticipated and unanticipated users.

For more information on UCore, including developer guides, tutorials, examples, and validation tools, see the Universal Core 2.0 site: http://www.ucore.gov (user registration required).

National Information Exchange Model (NIEM)

The NIEM represents a partnership of the U.S. Departments of Justice and Homeland Security. It is designed to develop, disseminate and support enterprise-wide information exchange standards and processes that can enable jurisdictions to share critical information effectively in emergency situations, as well as support the day-to-day operations of agencies throughout the nation. NIEM objectives include the following:

- Bring stakeholders and communities of interest together to identify information sharing requirements in day-to-day operational and emergency situations
- Develop standards, a common lexicon and an on-line repository of information exchange package documents to support information sharing
- Provide technical tools to support development, discovery, dissemination and reuse of exchange documents
- Provide training, technical assistance and implementation support services for enterprise-wide information exchange

For more documentation, training, and tools to support the NIEM, see the NIEM site: http://www.niem.gov.

Cursor on Target (CoT)

CoT is a data strategy for enabling DoD systems to exchange much needed time sensitive position or ***what***, ***when*** and ***where*** information. The CoT data strategy is based on a terse CoT XML Schema and a set of sub-schema extensions. The CoT schema is available on the DoD Metadata Registry [R1227]. Further CoT information is available at http://cot.mitre.org (user registration required).

Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM)

JC3IEDM is a data model developed by the Multilateral Interoperability Programme (MIP) Data Modeling Working Group. The aim of the MIP is to achieve international interoperability of Command and Control Information Systems (C2IS) at all levels. The MIP cooperates to develop a data modes that describe the information that allied component commanders need to exchange (both vertically and horizontally) and serve as the common interface specification for the exchange of essential battlespace information. The JC3IEDM is evolving from the Command and Control Information Exchange Data Mode (C2IEDM) data modeling efforts. Both data models are available on the MIP site http://www.mip-site.org [R1070]

Common Alerting Protocol (CAP)

CAP is a simple but general format for exchanging all-hazard emergency alerts and public warnings over all kinds of networks. CAP is developed and managed by **Organization for the Advancement of Structured Information Standards** (**OASIS**). CAP allows a simultaneous dissemination of consistent warning message over many different warning systems, thus increasing warning effectiveness while simplifying the warning task. CAP facilitates the detection of emerging patterns in local warnings of various kinds, such as might indicate an undetected hazard or hostile act, and CAP provides a template for effective warning messages based on best practices identified in academic research and real-world experience. The current version of the Common Alerting Protocol is available at http://www.oasis-open.org/specs/.

Naval Architecture Elements Reference Guide (NAERG)

NAERG is a key component of the coordinated set of activities intended to create a Department of the Navy (DON) Enterprise Architecture (EA). The NAERG supports the consistent and aligned development of architecture products across the DON, by implementing a common and reusable lexicon for naming the various elements within the federated DON EA. Further information see the NAERG site: https://sadie.spawar.navy.mil/Wiki/NAERG (DoD PKI Certificate required).

Guidance

- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1144: Develop two-level database models: one level captures the **conceptual** or logical aspects, and the other level captures the **physical** aspects.
- G1147: Use **domain analysis** to define the constraints on input data validation.
- G1148: **Normalize** data models.
- G1382: Be associated with one or more **Communities of Interest** (**COIs**).
- G1384: Review **XML Information Resources** in the **DoD Metadata Registry**, using those which can be reused.
- G1386: Review predefined commonly used **data elements** in the **Data Element Gallery** of the **DoD Metadata Registry**, using those in the **relational database** technology which can be reused in the Program.
- G1388: Use predefined commonly used database tables in the **DoD Metadata Registry**.
- G1390: Standardize on the terminology published by relevant **Communities of Interest** (**COIs**) listed in the **Taxonomy Gallery** of the **DoD Metadata Registry**.
- G1391: Identify **taxonomy** additions or changes in conjunction with the **Communities of Interest** (**COIs**) during the Program development for potential inclusion in the **Taxonomy Gallery** of the **DoD Metadata Registry**.

Best Practices

- BP1394: Identify, publish and validate data objects exposed to the enterprise early in the data engineering process and update in a spiral fashion as development proceeds.
- BP1397: Identify and develop use cases or reuse existing use cases as appropriate as early in the data engineering process as possible to support **data model** development.
- BP1398: Develop Interaction models as appropriate.
- BP1400: Programs will use authoritative **metadata** established by the Joint Mission Threads (JMTs) when available.

- BP1145: Use vendor-neutral **conceptual/logical models**.
- BP1396: Develop high-level conceptual data models for new systems prior to Milestone A based on the business process context in which the system will be used.
- BP1901: Use Universal Core (UCore) as the basis for information exchange models for systems that exchange internal data with external systems.
- BP1254: For **command-and-control** systems, use the names defined in the Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM) for data exposed to the outside communities.

P1049: Metadata

Services and **data** to be mediated should always be formally defined, and typically this is done with some form of computer readable **metadata**.

NESI currently requires metadata, defined primarily as **XML Schema** and **Web Services Description Language** (**WSDL**) documents, be registered in the **DoD Metadata Registry**. NESI further specifies rules system developers must follow in developing XML Schema, including the requirement to search the registry for existing schemas that can be reused, aligning new schemas as closely as possible to existing similar schemas, reviewing schemas with the DoD XML Namespace Manager, and looking for other relevant Government and industry schemas that could be leveraged. The purpose is to avoid unnecessary duplication of effort and improve the success of future interoperability through common definitions.

The NCES Data Strategy team, including the maintainers of the DoD Metadata Registry, strives to create a common data model, per **Community of Interest** (**COI**); but recognizing the difficulty in accomplishing that goal the team promotes the use of "mediation" from one schema to another. NCES currently implements mediation simply through the use of eXtensible Style Language Transformations (**XSLT**) to transform **XML documents** from one schema to another.

This focus on centrally managed data models is not viable as a long term solution to mediation since it requires substantial effort to define accurate transformations, and the underlying "business objects" almost always lose information in the process. The vision of a non-redundant object model is considered by most experts as unachievable due to social and communications barriers among the hundreds of organizations working as part of or with the Federal Government and the DoD in particular.

Accepting the fact that use of the DoD Metadata Registry is a requirement gives rise to posing the question should there be a new **FORCEnet** COI "**namespace**," or should the FORCEnet activities simply try to find suitable existing namespaces in which to register their metadata. Clearly, some FORCEnet applications will be able to leverage some of the existing schemas. But are there a significant number of new schemas to be registered, and if so can they be aligned to existing COI namespaces or will there be unacceptable barriers to introducing the changes required.

Moreover, the technologies for application and system development continue to improve to allow more rapid turnaround of new software capabilities, and in fact software developers are finding less of a need to work at the XML document level at all. **Model Driven Architecture** (**MDA**) technology, for example, is becoming mainstream, and **interfaces** are being developed visually, with the schemas automatically generated according to the graphical model. The creation of interfaces and schemas is becoming more of a dynamic activity, and the projected ad hoc interoperability of loosely coupled components, enforced by the FORCEnet vision, will mean bureaucratic processes such as those introduced by the DoD Metadata Registry may introduce significant risk.

Striving to minimize the number of schema variations by leveraging common schemas across applications is laudable and should be encouraged. However, more advanced solutions to mediation are critical to the interoperability problem where common schemas do not exist. This may require a more dynamic process for registering metadata, without restrictions. An argument can be made for a FORCEnet COI in this regard.

As promoted by the NCES Data Strategy team, XSLT is the common practice for mediation. However, XSLT only solves a single point-to-point integration, and it is limited in its ability to support semantic validation. The **Business Process Execution Language** (**BPEL**) is an emerging specification (likely to become a standard) for defining specific interactions among services using documents defined through schema. It can use XSLT and other technologies to perform transformation of data elements, and semantics are implicit through their use. However, each BPEL definition is limited even further to a single **use-case** for the data.

In order to reduce the work and the errors associated with mediation, we need to take the concept to the next logical step. Documents and services should include metadata that encodes their semantic intent. Technologies are emerging, such as the **Web Ontology Language** (**OWL**) [http://w3.org/], that assist in defining the semantic relationships and constraints in schemas.

These definitions can be used to automate the transformations between applications and services, to validate the transformations, and to support much more intelligent human-computer interaction. For example, a PEO C4I and Space sponsored program developed the Service Mediation Description specification for the DISA Net-Centric Capabilities Pilot. This metadata document automatically generated user interfaces (input forms, data result tables, and map overlays) from semantically-described **Web services** and schemas, using a document format—derived from BPEL and other Web standards.

Best Practices

- BP1408: Use a **semantic** description language such as **Web Ontology Language** (**OWL**) or **Resource Definition Framework** (**RDF**) to represent an **Ontology**.
- BP1409: Register **Web services** using **Web Services Description Language** (**WSDL**) and **Universal Description, Discovery, and Integration** (**UDDI**).
- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.

P1063: Relational Database Management Systems

A **Relational Database Management System RDBMS** is a collection of data items organized as a set of formally-described tables. This permits accessing and reassembling data in many different ways without having to reorganize the database tables. It is important to ensure data quality and to access data quickly, using simple, easily understood dynamic queries. Towards these ends, an **RDBMS** offers such services as **triggers**, **stored procedures**, indices, constraints, **referential integrity**, efficient storage, and **high availability** features.

Database Independence

The **Structured Query Language** (**SQL**) allows for some portability of database access code when accessing various database products. It is important to use SQL standards that are open and well supported by database vendors and to avoid using proprietary extensions to the SQL standards. To further promote database independence, access the database only through **open-standard** interfaces such as **Open Database Connectivity** (**ODBC**) or **Java Database Connection** (**JDBC**). This supports the goal of being able to swap out data sources and/or connect to multiple data sources without affecting the application or increasing software maintenance costs. Data-level adapters allow applications to access data through database calls that are native to the requesting application. At this point, the **business logic** can be shared with other data sources. This positions the application to move business logic from the database to the middle tier to support database independence.

Database Data Modeling

Data modeling is important for RDBMs as it improves database performance, improves the interoperability of the data, and allows for future growth and use of the RDBMS. The Data Modeling [P1003] perspective provides guidance for data modeling in addition to the guidance provided in this perspective.

Guidance

- G1146: Include information in the **data model** necessary to generate a **data dictionary**.
- G1153: Separate application, presentation, and data tiers.
- G1155: Use **triggers** to enforce **referential** or data integrity, not to perform complex **business logic**.
- G1151: Define declarative **foreign keys** for all relationships between tables to enforce **referential integrity**.
- G1154: Use **stored procedures** for operations that are focused on the insertion and maintenance of data.
- G1014: Access databases through **open standard** interfaces.
- G1141: Base new **data models** on existing data models developed by **Communities of Interest** (**COI**).
- G1144: Develop two-level database models: one level captures the **conceptual** or logical aspects, and the other level captures the **physical** aspects.
- G1147: Use **domain analysis** to define the constraints on input data validation.
- G1148: **Normalize** data models.
- G1151: Define declarative **foreign keys** for all relationships between tables to enforce **referential integrity**.
- G1132: Implement the data tier using **commercial off-the-shelf** (**COTS**) **relational database management system** (**RDBMS**) products that implement the **SQL** standard.

Best Practices

- BP1248: Follow a naming convention.
- BP1249: Do not use generic names for database objects such as databases, schema, users, tables, views, or indices.
- BP1250: Use case-insensitive names for database objects such as databases, schema, users, tables, views, and indices.
- BP1251: Separate words with underscores.
- BP1252: Do not use names with more than 30 characters.

- BP1253: Do not use the **SQL:1999** or SQL:2003 reserved words as names for database objects such as databases, schema, users, tables, views, or indices.
- BP1256: Use surrogate keys as the **primary key**.
- BP1257: Place a **unique key constraint** on the **natural key** fields.
- BP1260: Define a **primary key** for all tables.
- BP1261: Monitor and tune indexes according to the response time during normal operations in the production environment.
- BP1262: In the case of Oracle, define indexes against the **foreign keys** (**FK**) columns to avoid contention and locking issues.
- BP1263: Gather storage requirements in the planning phase, and then allocate twice the estimated storage space.
- BP1264: For **high availability**, use hardware solutions when geographic proximity permits.
- BP1254: For **command-and-control** systems, use the names defined in the Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM) for data exposed to the outside communities.
- BP1258: Explicitly define the encoding style of all data transferred via **XML**.
- BP1255: Use **surrogate keys**.
- BP1259: Use indexes.
- BP1140: Use SQL-2003 features in preference to **SQL-92** or **SQL-99**.
- BP1139: Do not use proprietary **SQL** extensions.
- BP1143: Use a **database modeling** tool that supports a two-level model (**Conceptual**/Logical and **Physical**) and **ISO-11179** data exchange standards.
- BP1145: Use vendor-neutral **conceptual/logical models**.
- BP1227: Do not allow installation of **MSMQ**-dependent clients.

P1133: Net-Centric Information Engineering

Of particular concern for **Global Information Grid** (GIG) interoperability is the information contained in inter-nodal information exchanges. Information exchanges are typically the purview of the systems within the Node, rather than the Node itself, and the details are worked out by a **Community of Interest** (COI). But the Node infrastructure must be engineered to support information exchanges between various COIs. The COIs can require any number of Components to fulfill the mission. When a Component wishes to make its data available to the **enterprise**, there are different enterprise design patterns the Component can use. For example, the mechanism selected by a Component to exchange information may be publish-subscribe, broker, or client server. The Node infrastructure must support whichever enterprise design pattern mechanism is selected. Consequently, the Node has a stake in the Component design. Additionally, the Node has a stake in performance specifications provided in the **Service Level Agreements** (SLA). The Node must support the SLA contract with the Node's infrastructure.

Node management should designate COI representatives to track, advocate, and engineer information exchanges in support of the **DoD Net-Centric Data Strategy**. According to this strategy, "COI is the inclusive term used to describe collaborative groups of users who must exchange information in pursuit of their shared goals, interests, missions, or business processes and who therefore must have shared vocabulary for the information they exchange." The principal mechanism for recording COI agreements is the **DoD Metadata Registry** required by the DoD CIO *DoD Net-Centric Data Management Strategy: Metadata Registration* memo. There are registry implementations on the **Non-Secure Internet Protocol Router Network** (NIPRNET), **Secret Internet Protocol Router Network** (SIPRNET), and **Joint Worldwide Intelligence Communications System** (JWICS).

The DoD Metadata Registry Web site (http://metadata.dod.mil) provides a search capability; there is also a **SOAP**-based interface to the Registry.

Guidance

- G1571: Maintain a comprehensive list of all the **Communities of Interest** (**COIs**) to which the **Components** of a Node belong.
- G1572: Include the Node as a party to any **Service Level Agreements** (**SLAs**) signed by any of the **components** of the Node.
- G1573: Define the enterprise design patterns that a Node supports.
- G1574: Define which enterprise design patterns a **Component** requires.
- G1575: Designate Node representatives to relevant **Communities of Interest** (**COIs**) in which Components of the Node participate.

Best Practices

- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.
- BP1866: Coordinate with end users to develop interoperable materiel in support of high-value mission capability.

P1329: Node Data Strategy

One of the key differentiators in the net-centric paradigm is the treatment of data as a key architectural element with particular attention on how data interoperates among different **Components**, **Nodes** and **Systems** in a net-centric enterprise.

The DoD ***Net-Centric Data Strategy (NCDS)*** [R1172] lays out specific approaches to achieve net-centric goals to provide visible, accessible, understandable, trusted and governable data. Common approaches allow Components and Nodes to handle data across multiple technical and organizational boundaries.

The Relationship to the DoD Net-Centric Data Strategy [P1299] perspective in Part 1: Overview [P1286] briefly describes the relationship between NESI and the DoD NCDS. The Net-Centric Data Strategy (NCDS) [P1204] perspective in Part 3: Migration [P1198] and the Data [P1244] perspectives supporting the ***ASD(NII) Net-Centric Checklist*** Data Tenets (P1244, P1250, P1252, P1253, P1254, P1256, P1257 and P1258 in NESI Part 2: Traceability [P1288]) contain detailed information including Guidance and Best Practices.

NCDS emphasizes developing community-based (versus enterprise-wide) data interoperability standards through collaborative governance forums known as **Communities of Interest** (**COIs**). DoD Directive 8320.2, ***Data Sharing in a Net-Centric Department of Defense*** [R1217] provides COI guidance in the light of achieving net-centric enterprise data goals. The Communities of Interest [P1302] perspective in Part 1: Overview [P1286] discusses how a COI shares a common vocabulary to exchange information.

For more detailed code level implementation information, see the set of perspectives related to Data [P1012] in the Part 5: Developer Guidance [P1118].

Relationship Between Data and Services

The DoD NCDS includes using services as a means of making any visible data accessible by the community or enterprise users. Such services could provide access either to mission data or to metadata describing the data or access to other available services or to their inventories. For example, a COI or a Program may choose to implement a utility service to transform or translate data.

Role of Node Infrastructure

Node infrastructure plays a key role in implementing a net-centric data strategy. It provides persistent information for data, as well as for any **metadata** that describes the data or the services available to access the data. Mission data access is not necessarily the same as metadata access; explicitly call out each interface, one a mission service and the other an infrastructure service. In other words, XML schemas, catalogs, etc., often live on a different server than the mission content. Node infrastructure also provides technological means of delivering data from the source to the consumer; e.g., using **Web** or messaging infrastructure on top of the underlining network to provide the conduit. The infrastructure delivers data via options including unchanged or transformed, within the Node or across Node boundaries, within the community or for the wider enterprise. Node infrastructure also provides all the necessary support and measures for the implementation of data security, management, fault tolerance and diagnostics.

Security Considerations

For security considerations related to data at rest see the Data at Rest [P1360] perspective in Part 5: Developer Guidance [P1118]. For security considerations for data in transit, see the Black Core [P1152], Confidentiality [P1340], Design Tenet: Encryption and HAIPE [P1247], and Public Key Infrastructure (PKI) and PK Enable Applications [P1061] perspectives.

Management Considerations

The DoD ***Net-Centric Data Strategy*** and the DoD ***Defense Information Enterprise Architecture (DEIA)*** [R1335] both address data management. The guidance in these references establishes metadata and schema registries and repositories which specify the structure of the data in question. The guidance also provides the overall governance and management processes for the registration and deposition of metadata and schemas

that makes the data visible and discoverable through directory services. The Security and Management [P1331] perspective contains additional related considerations on this topic.

Data management may also require managing multiple data registries and repositories, including federated configurations. One approach combines a locally-centralized Node data registry and repository with search or syndicated publication of data records in other registries and repositories.

Effective net-centric data management makes data visible, discoverable and accessible. Open standards such as **Extensible Markup Language** (**XML**) and ***Structure of Management Information*** (***SMI***; see RFC 2578) prescribe using metadata for specifying ordinary metadata, in turn (i.e., meta-metadata). Ensuring such standardized meta-metadata is common across all components, applications and services, helps component designers and architects understand the schemas and ordinary metadata, aiding data reuse so encoded from other components and services. In addition to making data visible, discoverable and accessible, metadata can establish data provenance and freshness through Data Stewardship processes.

In addition to these primary net-centric capabilities, data management includes configuration of content discovery and syndication that make data visible and discoverable through search or publication services.

It is often not possible to decouple the management of mission data often from management of the local computing infrastructure. Such computing infrastructure includes the file system or database and any associated user environment. Consider management of the local Web infrastructure when using **Web services** to expose the data and provide access.

Storage infrastructure management may have a major impact on mission data, since data challenges at the tactical edge often involve both storage and access to storage infrastructure. Management of databases and storage area networks goes beyond configuration; it also includes the necessary performance and fault management, such as in the following examples.

- ***Caching/Proxies/Distributed Masters***: use of content distribution constructs to deploy data closer to its consumers selectively
- ***High-Speed Transactions***: use of high-performance data storage constructs with transactional semantics to ensure producers and consumers are correctly synchronized

P1366: Data Management Services

This service area supports the administration of data independent of the processes that created it. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- DDS Data Local Reconstruction Layer (DLRL) [P1197]
- Relational Database Management Systems [P1063]
- Data [P1012]

P1367: Distributed Computing Services

This service area relates to distributed computing services to support applications that are physically or logically dispersed among computer systems in a network. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- Standard Interface Documentation [P1069]
- Implement a Component-Based Architecture [P1034]
- Public Interface Design [P1060]
- Messaging [P1047]
- Web Services [P1078]
- .NET Framework [P1086]
- CORBA [P1011]
- Data Distribution Service [P1190]
- Net-Centric Information Engineering [P1133]

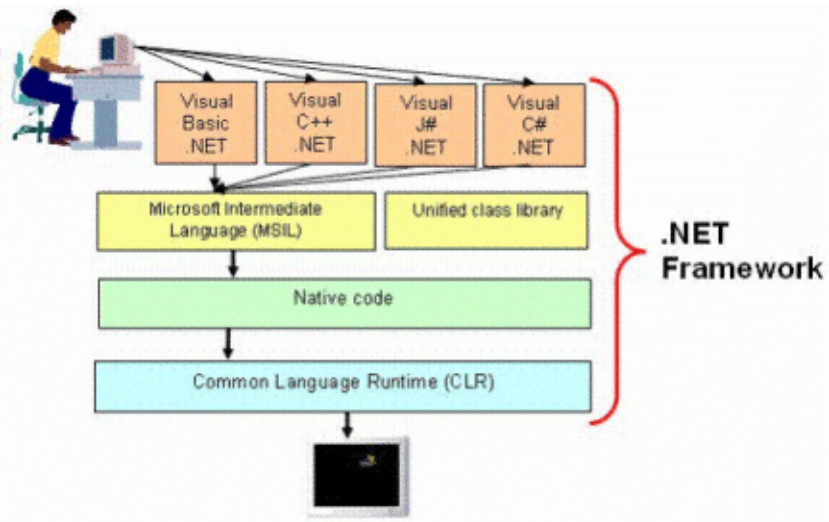
P1086: .NET Framework

To address the confusing maze of computer languages, libraries, tools, and toolkits that were necessary for creating multi-tier applications, Microsoft developed the **.NET** Framework and integrated it into Microsoft Windows as a **component**. It supports building and running multi-tier and **Service-Oriented Architectures** (**SOAs**), including **Web services** and **client** and **server** applications. It simplifies the process of designing, developing, and testing software, allowing individual developers to focus on core, application-specific code.

Microsoft summarizes the .NET Framework as

- A consistent, language-neutral, **object-oriented programming** environment.
- A code-execution environment that minimizes software deployment and versioning conflicts, guarantees safe execution of code, and eliminates the performance problems of scripted or interpreted environments.
- A consistent development environment.
- A framework composed of two key parts: the **Common Language Runtime** (**CLR**) and the **Unified Class Libraries**.

In the Microsoft .NET development environment, a programmer writes software in any one of several Visual .NET languages. These use a single, unified, object-oriented, hierarchical, and extensible set of class libraries to access the system and common services such as **XML** web services, enterprise services, ADO.NET, and XML. Next, the language source code is compiled into an intermediate **Microsoft Intermediate Language** (**MSIL**), which is later translated into platform-specific native code that uses the CLR.



I1064

Guidance

- G1210: For **.NET**, use Debug and Trace from the **`System.Diagnostics`** **namespace**.

Best Practices

- BP1097: Use the **`System.Text.StringBuilder`** class for repetitive string modifications such as appending, removing, replacing, or inserting characters.
- BP1098: Write all **.NET** code in C#.
- BP1100: Compile all **.NET** code using the .NET **Just-In-Time compiler**.

P1389: Enterprise Service Bus (ESB)

There are differing definitions within the computing industry and academia for the term **Enterprise Service Bus** (**ESB**). Some definitions describe an ESB as an **architectural style** or enterprise **design pattern** and other definitions describe an ESB as a middleware layer provided by a product or collection of products.

This perspective does not provide a new definition of ESB; rather, it explains ESB as an architectural style that provides distributed invocation, **mediation**, and end-to-end management and security of services and service interactions to support the larger architectural style known as **Service-Oriented Architecture** (**SOA**). In this perspective, as well as throughout NESI, the terms ***ESB*** and ***ESB architectural style*** are synonymous.

A common goal for implementing an ESB is to reduce **coupling** in service interactions by providing architectural components which act as intermediaries to provide mediation and service virtualization. This reduced coupling provides for a clean separation of concerns in areas such as implementation technologies and standards, transport protocols, design and messaging patterns, configuration management, personnel (to include developers, administrators, and operational support personnel), and organizations.

> ***Note:*** *This definition of an ESB as an architectural style does not preclude vendors from providing solutions that implement the ESB architectural style, nor does it prevent one from calling an ESB implementation an Enterprise Service Bus.*

The ESB architectural style requires the hosting of services. Without services, the resulting architecture would be nothing more than **Message Oriented Middleware** (**MOM**) or a message broker. Implementing these services does not necessarily requires the use of SOAP; the ESB architectural style often exposes many types of service implementations such as services based on Representational State Transfer (REST) or **Java Message Service** (**JMS**).

The ESB architectural style leverages the concept of a bus as a subsystem that transfers data between endpoints. Traditionally, without the use of an ESB, the service provider and the consumer engaged in an interaction must agree on the same protocol and message format. In essence, each protocol and message format becomes its own bus.

In contrast, an ESB implementation behaves as a universal bus by providing adapters that allow service providers and service consumers to interact without concern for the specific protocol and format of each other. The end result is that the provider and consumer are less coupled (for example in protocol, location, and message format). Each is still coupled to an underlying protocol and format that are usually based on open standards. For example, a service consumer that wants a service delivered using HTTP can easily interact with a service provider that offers services using JMS.

An ESB generally has core characteristics in the areas of services, invocation, messaging, mediation, transport, management, and security as shown in the table below.

Services	Support to host and manage services
Invocation	Support for consumers to locating and binding to services
Messaging	Support for service providers and consumers to communicate through the exchange of well-defined messages through various communication patterns to include synchronous, asynchronous, and publish and subscribe
Mediation	Support for transformation, aggregation, adaptation, orchestration, and choreography. Mediation may occur on many areas to include message content, transport protocol, **quality of service** (**QoS**) parameters, service version, etc.
Transport	Provides for routing, transport, security, and guaranteed delivery of message between service providers and service consumers, often through the use of message routers and adapters for various standards based communication protocols
Management	Support for the management of service interactions and status to include, alerting, auditing, logging, QoS monitoring, configuration management, and metric collection

Security	Support for enforcing enterprise security polices and adapting to security threats

In addition to these core characteristics, an ESB generally provides the following capabilities:

- An ESB allows for the service providers to provide data at a rate independent from the consumer's consumption rate. ESB implementations often supports the pairing of consumer and providers based on QoS parameters and by providing message filtering capabilities.
- An ESB provided an opportunity for service providers to compartmentalize their implementations behind a well-defined interface so that consumers can use the service without having to understand the internal details of the service.
- An ESB enables loose coupling of service providers and consumers which aids integration and composeability. Service consumers are blind to implementation technologies used by service providers and vice versa. Any number of service providers may process a request message dynamically based on QoS or location. An ESB provides support for late binding of service endpoints. Consumers and providers do not have to agree on transport protocol or endpoint addresses.
- An ESB support service versioning by isolating changes to services. Service consumers can continue making request to older versions of a service while an ESB provides mediation services.
- An ESB reduces the number of point-to-point contacts between service providers and service consumers easing integration and making impact analysis for changes or vulnerabilities easier.
- An ESB provides service logging to include what services are used, who uses them, the performance of the service interactions, and exceptional conditions and errors.
- An ESB supports fault tolerance through concepts such as intelligent routing , redundant service providers, and execution of **Business Process Execution Language** (**BPEL**) to support recovery.
- An ESB supports composition and execution services to support business processes to include long-running transactions. This is usually done through the use of BPEL.
- ESB implementations are aided by existing developer and engineer skills with technologies such as **XML**, XML Path Language (XPATH), and eXtensible Style Language Transformations (XSLT).
- An ESB is an enabler for reuse by allowing for expose legacy systems through the use of adapters resulting in a possible cost savings.
- An ESB helps manage risk through incremental SOA implementation.
- An ESB Supports distributed SOA implementation.

Although an ESB may provide many advantages for SOA implementation, several challenges remain:

- There is not an industry-wide agreed upon definition for ESB and there is not a single ESB standard. As a result, vendors support various capabilities within their ESB support products which can lead to vendor dependence and coupling.
- An ESB infrastructure may increase latency between service consumers and service providers compared to a direct stovepipe connection.
- An ESB infrastructure can become a major point of failure in a system as well as a major target for penetration of denial of service attacks.
- Mapping between information exchange patterns may not be optimal.

The following general guidelines, in addition to formal NESI guidance, may help to mitigate these concerns.

- Content providers should be responsible for translations, not the ESB since it forces the ESB development team to have a detailed understanding data models and interfaces of service providers and service consumers.
- Do not implement an ESB until you need one, and only implement one once you have a SOA strategic vision and a set of adoption project plans. An ESB is a means to and end and not an end in itself. Delaying and ESB implementation will save resources until such time they are needed an allow time for industry to mature standards and tools for implementing the ESB.
- Adopt and Implement an ESB incrementally to build upon lessons learned.
- Provide a common set of management capabilities for services and endpoints including alerting, statistics, audits, and logging for an ESB.

- Design and implement an ESB to scale beyond the performance requirements of all service providers and consumers deployed within the ESB. XML performance for streaming data and transformation is particularly important. Non-blocking input and output is also required to prevent components from blocking while waiting for other components to respond.
- Design and implement an ESB to s support the overall enterprise security policies for the relevant organizations by incorporating controls for overarching SOA security policies.

Guidance

- G1910: Provide for transformation of **XML** messages using **eXtensible Style Language Transformations** (**XSLT**) when implementing an **Enterprise Service Bus** (**ESB**).
- G1912: Support the execution of **Business Process Execution Language** (**BPEL**) when implementing an **Enterprise Service Bus** (**ESB**).

Best Practices

- BP1908: Provide bidirectional mediation between transport protocols mandated in the **Defense IT Standards Registry** (**DISR**) when implementing an **Enterprise Service Bus** (**ESB**).
- BP1909: Provide for filtering of **XML** messages using XML Path Language (XPath) when implementing an **Enterprise Service Bus** (**ESB**).
- BP1911: Provide for routing of messages based on message content when implementing an **Enterprise Service Bus** (**ESB**).
- BP1913: Provide for mediation between synchronous and asynchronous messages when implementing an **Enterprise Service Bus** (**ESB**).

P1368: Environment Management

This service area relates to data processing and communications environment management. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- Implement a Component-Based Architecture [P1034]
- Public Interface Design [P1060]
- Software Communications Architecture [P1087]
- Enterprise Management [P1330]
- Standard Interface Documentation [P1069]
- Services [P1164]

P1330: Enterprise Management

Enterprise Management involves planning, organizing, staffing and governing an **enterprise**. A Node packages operational capabilities into standard technology-based components (see the NESI Node Decomposition [P1343] perspective). Each component, regardless of functional area, has management information associated with it that makes it manageable throughout its lifecycle while at the same time enabling their assembly into a Node within the lifecycle and operational context of that Node. This management information is available to authorized managers through management interfaces (to include paper and electronic means).

In addition to a technical Node decomposition viewpoint, there is a semi-standardized Lifecycle decomposition viewpoint that the business operations community of the enterprise management generates. The community that manages infrastructure service operations (often referred to as **NetOps**) further focuses on aspects of the Node and Lifecycle viewpoints in a more detailed activity decomposition view.

Thus, the following three viewpoints, each with applicable standards and governance, may apply when considering or decomposing enterprise management functions.

- ***Component*** - identifies guidance and necessary interfaces to manage the Node components throughout the lifecycle
- ***Lifecycle*** - identifies guidance about configuration management, change management and responsibility handoffs
- ***Operational Activity*** - identifies detailed guidance for the deployment and operational support phase of the lifecycle

Component Viewpoint

Three basic principles help describe Enterprise Management:

- ***Decomposition*** breaks the enterprise down into modules for management purposes
- ***Delegation*** assigns the responsibility for managing each module to a representative management agent; delegation of responsibility may be applied through a tiered approach such that hierarchies of management agents may aggregate, collate and correlate management information reported by more localized management agents
- ***Decision authorization*** specifies where, when and which policies and human oversight affect Node and component operations, including machine-to-machine operations; decision authority in machine-to-machine operations rests in policy decision points and policy enforcement points, which may be in separate component modules (often the manager and agent, respectively) or co-located due to performance or security constraints

Standards, in addition to the above principles, play an important role in enterprise management. For interoperability and enterprise management purposes, each type of managed module must identify itself and publish a standardized version of the management information and operations it makes available to enterprise management systems. For example, a managing component may interact remotely with the modules it is responsible for managing. In this case, each module will reside on a network and use standard transport interfaces and management protocols such as the ***Simple Network Management Protocol*** (***SNMP***). To enable management functions, each instance of a managed module must have a **Uniform Resource Identifier** (**URI**) that enables deploying, provisioning, monitoring and adjusting in accordance with the enterprise's policies and protocols. Management URIs are usually defined as part of the data standard's protocol. For example, STD 62 (IETF RFC 3418) uses SNMP **URLs** for management URIs.

Lifecycle Viewpoint

Traditionally, lifecycle decomposition is a procedural decomposition of change management. Since responsibility and authority for controlling change is a jealously guarded right of every organization, no matter how small, a standard lifecycle decomposition must enable customized and tailored components while simultaneously establishing minimum acceptance and interoperability criteria of those components.

Historically, coordinated change management between organizations (including acceptance and interoperability testing) was either not necessary due to independent organizations without interaction or routinely was built-in as a unified command or other overarching higher authority that aligned subordinates. In either case, the result was a single change management process: either a relatively simple local process, or highly political deconfliction interactions between high level leaders. Consequently, there were no successful open international standards because they poorly replicated existing processes at a higher overhead cost.

This situation is changing; with the rise of software and its inherent dynamic and complex configuration management, developers felt the need for more formal standards of acceptance and interoperability, one amenable to industrialized production methods. There have been several attempts to promulgate these standards, such as the International Organization for Standardization "Quality management systems - Guidelines for configuration management" (ISO 10007, 1 June 2003) and "IEEE Standard for Software Configuration Management Plans" (IEEE Std 828). However, due to the extreme diversity of software products, classic methods of industrialized production have not brought many of the anticipated cost reductions, and none has seen widespread adoption as a unified standard. A number of common concepts, constructs, and procedures are emerging and do show up in various standards optimized for a particular Node decomposition area. They first appeared in the Transport area, in the **Internet** standards venue, when the **Internet Engineering Task Force** (**IETF**) standards body insisted on cross-organization interoperability as the primary criterion for acceptance of any proposed standard.

The U.S. Government describes the lifecycle procedural breakdown with a major emphasis on acquisition and a minor one on operation. Thus, Lifecycle decomposition is driven by two basic principles:

- a spiral of change in which distinct organizational roles hand-off responsibility for change management of a system, component or Node
- a minimal set of process constructs: management roles, protocols and data that serve to coordinate the handoffs and provide continuity throughout the spiral

Additionally, two things drive the elaboration, refinement or extension of these principles:

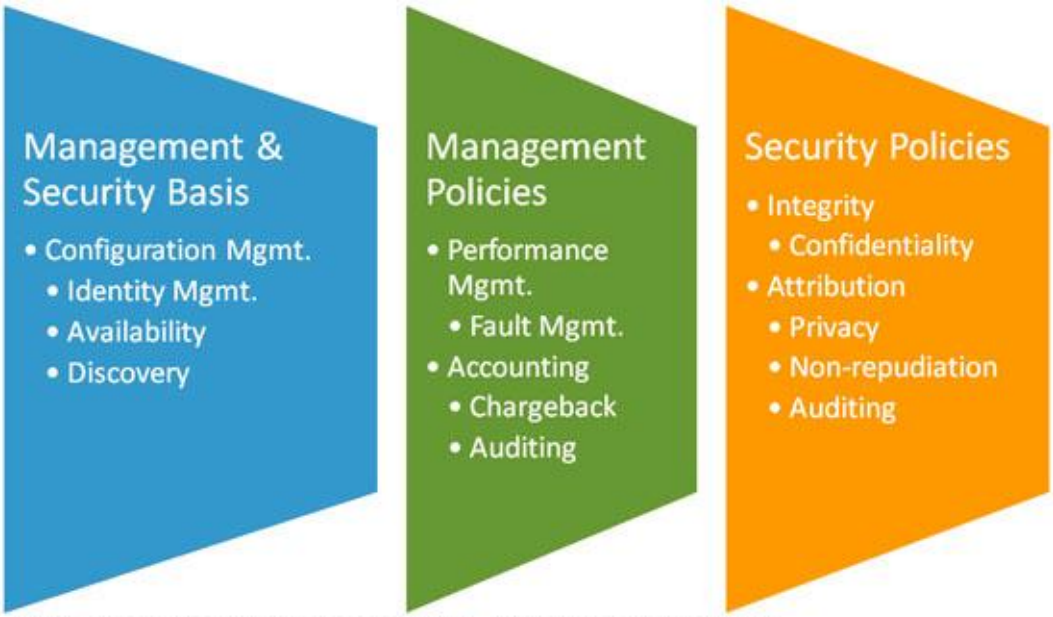
- the resources available to the organization for refinement of process constructs
- the resources required for development and integration of replacement technology in accordance with the refined process

Operational Activity Viewpoint

A refinement of the Component and Lifecycle Viewpoint decompositions into the Transport, Networks and Telecommunications functional areas generated the initial Service Operations Activity Viewpoint decomposition. Five areas defined the original decomposition: ***Fault***, ***Configuration***, ***Accounting***, ***Performance*** and ***Security***; thus, this decomposition is known by the acronym ***FCAPS***. The standards body which has evolved into the **International Telecommunication Union** first developed this reference framework for telecommunications management, captured by the ISO X.700 family of standards which is now part of the ITU-T Recommendation series M.3000.

The five activity areas were originally seen as independent; as the standard developed, it became evident that they were sufficiently inter-dependent that a single protocol was sufficient to cover all five areas. The main differences among the activity areas were how human oversight and policies were included. Configuration Management (to include Identity Management, Availability and Discovery) is the foundation layer for both Management and Security; the relevant policies are simply statements of the acceptable bounds of existence (what is in the configuration inventory) and efficacy (which types, versions, and default behavior options).

The split between Management and Security derives from the different types of operational and organizational policy drivers: efficiency and assurance. Management of efficiency drives Performance Management plus its extension, Fault Management, and its organizational policy management, Accounting (to include Chargeback and Auditing). Security (responsibility for assurance) drives Integrity plus its extension Confidentiality, and its organizational policy management Attribution (and its extensions Privacy, Non-Repudiation and Auditing). Management (efficiency) and Security (assurance) policies are generally captured in a relevant profile or other policy construct.



I1240: Operational Activity Decomposition of Enterprise Management

Note that such profiles must be appropriate to the Node Operating Environments [P1345] in which they are deployed, in accordance with operational guidance, with the following characterizations:

- Configuration includes component type, count and rate of change for making effective selections over the whole portfolio and lifecycle (i.e., development, production and deployment, operations and support).
- Management includes component resource availability, expected capacity and rate of consumption and expected level of tolerable inefficiencies.
- Security includes components' tolerance of change (especially unexpected, unauthorized and enterprise management changes) and assurance of sufficient efficiencies to provide resource reserves necessary for resilience and expected levels of interference and threats

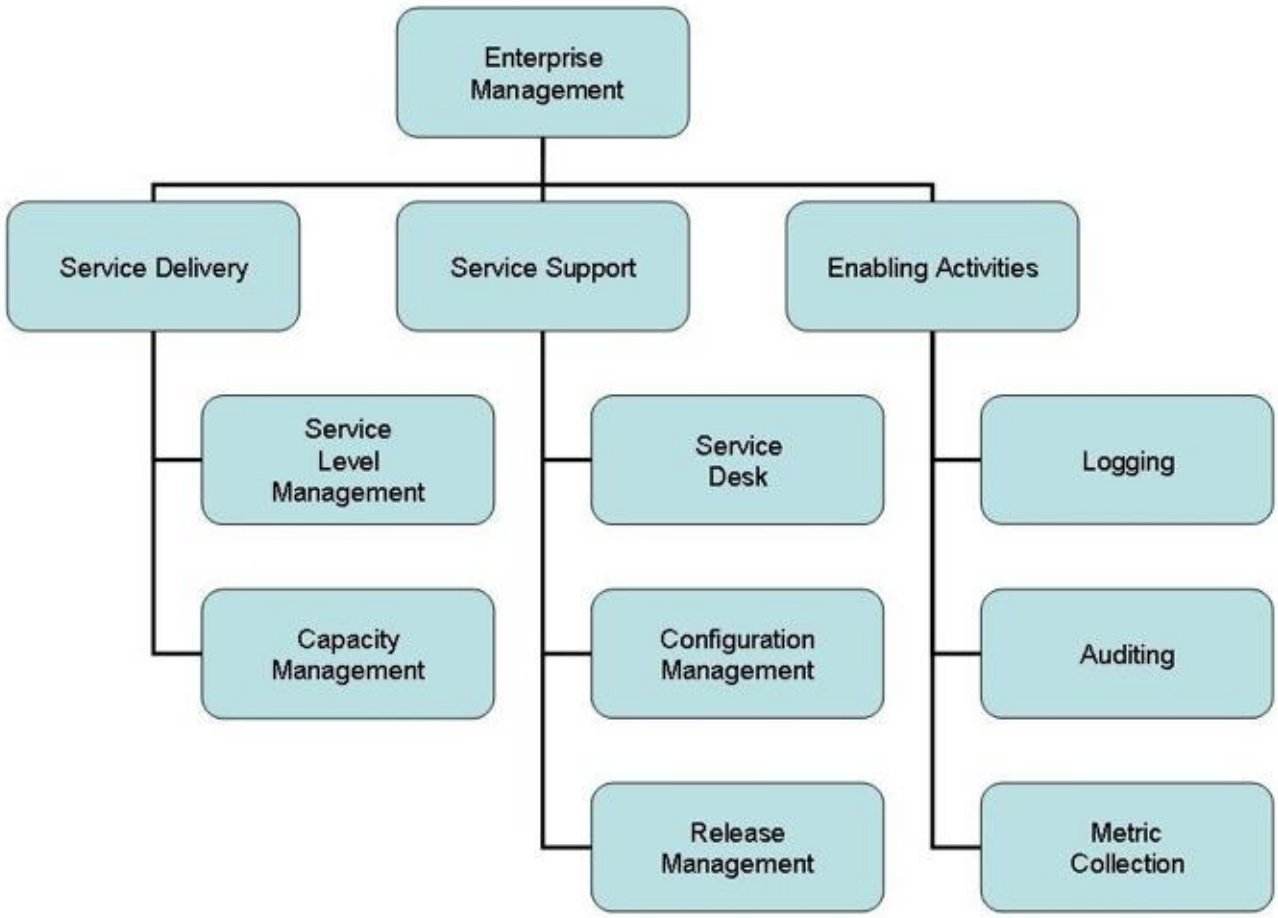
Enterprise Management Decomposition Example

The *Example Enterprise Management Decomposition* diagram below (I1219) illustrates a decomposition of enterprise management into service delivery, service support and enabling (supporting) activities.

- ***Service Delivery*** monitors and reconfigures the provisioned capabilities and capacities according to dynamic policy needs
- ***Service Support*** covers the selection, identifier assignment, deployment, default provisioning, and default configuration of managed entities in accordance with the enterprise's planned operations and policies
- ***Enabling Activities*** support both service delivery and service support

Service organizations may stand up a variety of functional teams that focus on planning and deployment, provisioning, configuration and report analysis, and monitoring and incident handling, with manager systems equipped for information fusion, operations coordination, analyses, report generation, planning and policy creation.

Beyond the simpler task of maintaining status information such as link status or service up/down status, enterprise management may include complex service arrangements involving multiple, orchestrated services. Additionally, coordinated help-desk support and reporting are needed. The DoD NetOps concept is addressing some of these topics.



I1219: Example Enterprise Management Decomposition

The following subsections describe in more detail the Service Delivery, Service Support and Enabling Activities modules in the *Example Enterprise Management Decomposition* diagram (I1219).

Service Delivery

Service Level Management

Service Level Agreements (**SLAs**) specify performance requirements, measures of effectiveness, reporting, cost, and recourse in a contractual agreement between service providers and consumers.

Capacity Management

This aspect of service delivery manages the ability to provide services in order to meet the level of performance specified in SLAs. Faults of various kinds can disrupt service delivery capacity and thus require active management.

Fault Management

Fault management constitutes the activities of identifying, analyzing and handling faults; in other words, recognizing when performance is so out of expected or relied upon range that policy dictates reaction. Performance metrics collected as part of the Enabling Activities provide data to support analysis. Fault management is the process of defining threshold policy constructs that cover unacceptable behavior (refer to the Enterprise Security [P1332]-related perspectives for additional information), starting with unacceptable performance. The two (unacceptable behavior and unacceptable performance) overlap in their common need for enabling technologies such as standardized threshold policy constructs, event logging and in those policies surrounding availability when poor performance can constitute a denial of service attack.

Service Support

Service Desk

A Service Desk contributes to service support by monitoring and responding to situations which impact performance, integrity, faults, accounting and attribution aspects of service delivery.

Incident Management

An incident is any event which is not part of the standard operation of the service and which causes, or may cause, an interruption or a reduction of the quality of the service.

The objective of incident management is to restore normal operations as quickly as possible with the least possible impact on either the business or the user at a cost-effective price.

Inputs for incident management mostly come from users, but inputs can have other sources as well such as management information or detection systems. The outputs of the process include change requests, resolved and closed incidents, management information, and communication to the customer.

Problem Management

Problem Management is the process responsible for managing the life cycle of all problems. The primary objectives of problem management are to prevent incidents from happening and to minimize the impact of incidents that cannot be prevented.

Configuration Management

Configuration Management relies on the persistent and continually updated storage of information about the elements that an organization uses in the provision and management of its **information technology** (**IT**) operations and management. Classically, this information base is implemented as a database; hence, the Information Technology Infrastructure Library (ITIL) term ***Configuration Management Database*** (***CMDB***). This is more than just an asset register, as it usually contains information that relates to the maintenance, movement, and problems experienced with Configuration Items (CIs).
The CMDB also holds a much wider range of information about items upon which the organization's operations and management depend to include the following:

- ***Hardware***
- ***Software***
- ***Documentation***
- ***Personnel***

Configuration Management essentially consists of four tasks:

- ***Identification*** - the specification, identification of all IT components and their inclusion in the CMDB
- ***Control*** - the management of each CI, specifying who is authorized to change it
- ***Status*** - the recording of the status of all CIs in the CMDB and the maintenance of this information
- ***Verification*** - the reviews and audits to ensure the information contained in the CMDB is accurate

Without the definition of all configuration items that provide an organization's operations and management, it can be very difficult to identify which items are used for which services. This could result in critical configuration items being stolen, moved or misplaced, affecting the availability of the services dependent upon them. It could also result in using unauthorized items in the provision of operations and management.

Note: *Configuration Management (CM) does not require a database, which is a particular architectural choice. CM in network and Web environments is often done with either directory service registries or search-based discovery services, and the results are not necessarily stored in a database.*

Assets and Resources

The essence of configuration management is to inventory and identify a Node's technology and information component assets and group them into recognized operational assets. Assets come in many types and each service concentrates on those in support of particular concept of operations (CONOPS). The Air

Force, for example, recognizes the following asset categories (refer to Air Force Doctrine Document 2, *Operations and Organization*, 3 April 1997 and AFDD 2-5.1, *Electronic Warfare*, 5 November 2002):

- Command and Control (C2) and Force Protection
- Intelligence, Surveillance and Reconnaissance (ISR)
- Inter- and Intra-theater Air Mobility
- Air and Space
- Electro-magnetic Spectrum Control

Information System assets are less obviously traceable; however DoD Directive 3020.40, *Defense Critical Infrastructure Program (DCIP)*, 19 August 2009 specifies any distinguishable network entity that provides a service or capability as an infrastructure asset.

Hardware historically has been the basis for managing assets (as materiel or facilities). Increasingly, however, infrastructure and mission software and services are becoming distinguishable assets and defining them as infrastructure simply because their network hardware address distinguishes them is increasingly insufficient. Net-centric operations and service-oriented approaches have demonstrated the limits of treating software in much the same way as hardware and treating the shrink-wrapped package as the asset instead of the capability the software provides.

Identifiers

Uniform Resource Identifiers (URIs) are a basic pre-requisite to Node manageability. Identifiers often provide more than a distinguishing attribute; they often overload the identifier with metadata about the named entity's functional decomposition (as in structured identifiers). Using a particular naming authority (for example, mailto), clarifies the requisite Transport and other Node decomposition infrastructures. For example, the mailto authority defines the user environment rendering of email messages, the computing infrastructure processing and storage data types, and optionally, the cryptographic infrastructure encoding information to expect.

Asset Types and Metadata

Overloading an identifier with all possible current and future metadata about an asset's type, especially when the asset types were produced under multiple authorities, proved infeasible and to the creation of an easily extendable standard framework for specifying standard management metadata, the Common Management Information Service (CMIS). This particular encoding was too processing intensive and essentially has been replaced by the simpler tabular encoding of the SNMP Structure of Management Information (SMI) approach. Subsequently, the ASN.1 protocol encoding of both CMIS and SMI became so optimized as to make it unmanageable by humans. This led to a proposal to use the more readable XML encoding of the Distributed Management Task Force (DMTF) Common Information Model (CIM) instead. The NetOps community deemed that the XML performance was too poor; BinaryXML encoding of the management information model and protocol is currently under discussion for both SNMP SMI and DMTF CIM protocols. Consequently, typed asset identifiers for software packages are still used in common practice. See the Java EE Deployment Descriptors [P1037] perspective for a detailed discussion and recommendations of one such use case. Attention to interoperability between computing infrastructure structured, type-encoded identifiers such as file extensions and Management identifiers such as XML strings will pay off in seamless management operations.

Asset Types and Unique IDs

All asset identifiers must provide the ability to distinguish an asset from any other asset within the management domain. Since the size and population of that management domain cannot be determined except in the field, asset identifier size requirements must be sufficiently large to provide a suitable namespace and mechanisms to extend that space if necessary. In addition, political authorities structure the global asset namespace, starting at the global level with the Internet Corporation for Assigned Names and Numbers (ICANN);[R1314] this is most evident in the allocation and assignment of unique instance IDs. Finally, asset management systems must be sized to cover and support the potential inventory types and total number of instances.

Versioning

Version identifiers are also necessary, given that assets may evolve over time without substantially changing capability or deployed role while changing in at least some sufficiently important particular. Unfortunately, this automatically sets up a potential conflict between component vendors who wish to highlight each improvement for marketing purposes and configuration and change management personnel who wish to minimize the amount of acceptance interoperability testing. The latter community has attempted to provide version numbering standards, but they are best practices and often limited to particular component types.

Change Control Management

Change control management uses a formal process to ensure that the introduction of changes to a system is in a controlled and coordinated manner. This process includes assessing all changes for risks and assessing the potential business impacts should a change produce undesired results.

If change control management procedures are not effective, unauthorized changes to operations and management may result. This could have major business impacts, including financial loss, customer loss, market loss, litigation, and in the worse case scenario, even collapse of the business that the operations and management are there to support.

In addition to change management of versioned releases and their patches, the configuration change management community distinguishes between deployment and provisioning, in order to separate the processes centered around hardware acquisition and physical configuration from the processes centered around enabling, activating and other software-based configuration changes, respectively.

Deployment

Deployment generally refers to those management activities, processes and data concerned with acquisition, especially capital expenditure governance, and physical installation and configurations.

Provisioning

Provisioning generally refers to those management activities, processes and data concerned with allocation and assignment of infrastructure, shared or common resources, especially the accountability, charge back and customer management aspects, and virtual asset configurations.

Software Asset Management

Software Asset Management (SAM) is the practice of integrating people, processes and technology to allow software licenses and usage to be systematically tracked, evaluated and managed. The goal of SAM is to reduce IT expenditures, human resource overhead and risks inherent in owning and managing software assets.

SAM includes maintaining software license compliance; tracking the inventory and usage of software assets; and maintaining standard policies and procedures surrounding the definition, deployment, configuration, use and retirement of software assets. SAM represents the software component of IT asset management, but SAM also is intrinsically linked to hardware asset management by the concept that ineffective inventory hardware controls significantly inhibit efforts to control the software thereon.

Patch Management

Patch Management is an area of systems management that involves acquiring, testing, and installing multiple patches (code changes) to an administered system. Systems can include servers, routers, personal digital assistants (PDAs), etc. Patch management tasks include maintaining current knowledge of available patches, deciding what patches are appropriate for particular systems, ensuring that patches are installed properly, testing systems after installation, and documenting all associated procedures, such as specific required configurations.

Patches sometimes are ineffective and can cause more problems than they fix. System administrators can take simple steps, such as performing backups and testing patches on non-critical systems prior to installations, to avoid problems caused by unintended side effects of patches.

Release Management

Release Management is the process that encompasses the planning, design, build, configuration and testing of hardware and software releases to create a defined set of release components. Release activities also include the planning, preparation, scheduling, training, documentation, distribution and installation of the release to many users and locations. Release Management uses the controlling processes of Change and Configuration Management.

Enabling Activities

Logging

Log Management

A log is a record of the events occurring within an organization's systems and networks. Logs are composed of log entries; each entry contains information related to a specific event that has occurred within a system or network. Originally, logs primarily supported troubleshooting problems. Logs now serve many functions within organizations, such as optimizing system and network performance, recording the actions of users, and providing data useful for investigating malicious activity. Logs have evolved to contain information related to many different types of events occurring within networks and systems. Within an organization, many logs contain records related to computer security; common examples of these computer security logs are audit logs that track user authentication attempts and security device logs that record possible attacks.

Audit Log

An Audit Log is a record of transactions in an information system that provides verification of the activity of the system. The simplest audit trail is the transaction itself. For example, if a person's salary is increased, the change transaction includes the date, amount of raise and name of authorizing manager.

A more elaborate audit trail can be created when the system is verified for accuracy; for example, samples of processing results can be recorded at various stages. Item counts and hash totals verify that the system has processed all inputs.

An audit trail can include any activity whatsoever, but transactions that do not effect a change are often not recorded. For example, ad hoc searches and database look-ups may not be identified in an audit trail, and routine queries are typically exempt from auditing.

Auditing

Every operating system (OS) includes security features and vulnerabilities which vary from OS to OS and sometimes between versions of the same OS. The security features are designed in such a way that they can be turned on or off and set to high security or low security, depending on the purpose for which the user intends to use the OS. In most cases, the default settings are not designed for high security. It often is up to the user to enable the security features to the desired level of security for that installation.
The process of auditing OS security includes evaluating whether the security features have been enabled and the parameters have been set to values consistent with the security policy of the organization and verifying that all users of the system (user IDs) have appropriate privileges to the various resources and data held in the system.

Metric Collection

Collection of metrics is a prerequisite for good performance analysis. Metrics are a key component in enabling functionality for the modules in the Example Enterprise Management Decomposition figure (I1219) included in this perspective. Multiple open standards define common infrastructure metrics for many categories such as in the following examples:

- Transport metrics defined as part of a component's Management Information Base (MIB) counters, for example RFC 2863 interface counters
- Various specification benchmarks define computing infrastructure metrics

Performance Metrics

Node and component performance, both infrastructure and mission-oriented, have an impact on net-centric operations. In a dynamic environment, where information exchange sources may not be infrastructure service providers, infrastructure metrics can be a key factor in the selection of service and information sources. Performance metric metadata, when advertised externally and frequently updated, allow potential service users to compare and select an implementation that meets their performance requirements, such as a measurement of reliability. Metrics are needed also to determine if performance has been supplied according to more traditional Service Level Agreements and for common infrastructure operations management.

Standard instrumentation for the collection of performance metrics of Nodes and components is necessary for management interoperability. Metrics should be visible and accessible as part of component service registration and updated periodically. See the Instrumentation for Metrics [P1163] perspective for more detailed information.

Performance Parameters and Ranges

Performance metrics are constituted from a combination of the base parameter type and its nominal (native default) range of values, for example a process execution counter. Simply collecting and monitoring such metrics may be sufficient for simple performance management; such metrics are so common as to be the default in the management information constructs such as SNMP MIBs and the DMTF CIM. In larger or more complex systems, performance metrics may include policy constructs that define the expected and reasonable ranges of performance parameters and increment, for example, a high- or low-watermark counter when exceeded, to aid in future capacity planning and even immediate adaptation activities.

Fault Thresholds and Policies

When the nominal or expected range of a performance parameter is far exceeded or exceeded for an unduly long time, most components management information models include thresholds: policy constructs that define alert or alarm events. In addition, there is an enabling event and logging infrastructure that generates event messages, sends them to the appropriate management system for logging, correlation, analysis, and potentially triggers corrective or adaptive reactions.

Web Service Metrics

Descriptions of some sample metrics that may be appropriate for **Web services** are in the Instrumentation for Metrics [P1163] perspective.

Best Practices

- BP1688: For **Services Management**, use an interim solution based on standardized Simple Network Management Protocol (SNMP) agents or other locally provided instrumentation and external monitoring tools.

P1164: Services

The *DoD Net-Centric Services Strategy* (NCSS) [R1313] establishes **services** as the preferred means by which data producers and capability providers make their data assets and capabilities available across the Department of Defense (DoD) and beyond. The DoD vision is to establish a Net-Centric Environment (NCE), a framework for human and technical connectivity and interoperability. This environment allows DoD users and mission partners to share and protect information, to make informed decisions, and to leverage shared services and **Service-Oriented Architecture** (**SOA**) that have the following characteristics:

- Supported by the required use of a single set of standards, rules, and a common, shared secure infrastructure provided by the Defense Information Enterprise Mission Area (DIEMA)
- Populated with appropriately secure mission and business services provided and used by each mission area
- Governed by a cross-Mission Area board, chaired by the DoD **Chief Information Officer** (**CIO**)
- Managed by **Global Information Grid** (**GIG**) **Network Operations** (**NetOps**).

Service-Oriented Architecture (SOA) is an architectural style for describing an environment in terms of distinct shared mission and business functions and data exposed as carefully designed, available, secured and managed services. Such services, therefore, are often referred to as "mission" or "business services" and they usually reside in the application layer of the architecture (where the mission and business applications typically reside). Since each carries a distinct mission or business function, they serve as building blocks for key elements of mission or business functionality that can become mission threads and business flows.

Services built specifically for the purpose of creating accessibility for visible mission data and metadata, as part of the *DoD Net-Centric Data Strategy* [R1312] implementation, are also part of the enterprise. As described in the Node Data Strategy [P1329] perspective, some of those data services potentially may be used in operational environments as described above. This would depend on the specific need for the exposed data, maturity level of the service, service ownership, and other factors.

Carrying a business or mission value is not the only characteristic of a service upon which the SOA architectural style is built. One other characteristic of a service is implementation in a loosely coupled manner that, in some cases, would allow orchestrating the service into flows even at run time, creating services composed of other services, and changing the internal implementation of a service without affecting its interface. See the Service-Oriented Architecture [P1304] perspective in NESI Part 1: Overview [P1286] for a list of distinct characteristics that identify a service in SOA. See also NESI Part 3: Migration Guidance [P1198] for discussions on SOA migration of legacy systems and SOA maturity levels.

Another key component of the DoD services vision is the establishment of the enabling and execution environment for mission/business services. This support environment consists of the following:

- Infrastructure responsible for the reliable, timely and secure delivery of service execution results to the consumer
  - Hardware, Operating Systems
  - Networking [P1138]
  - Data storage
  - Middleware that may include Web Infrastructure [P1157], Message-Oriented Middleware [P1046], data servers (e.g., **RDBMS**), run-time service discovery, etc.; some of the middleware-related topics are also discussed in the Information Exchange Patterns [P1326] and Service Optimization and Scalability [P1327] perspectives
  - Utilities and functions responsible for resolving interoperability and integration issues for seamless services communications within or across management boundaries; see the Utility Services [P1328] perspective regarding commonly used techniques
  - Security and Management [P1331] measures implemented within all of the above elements and as specialized utility applications
- Services and functions, along with their underlying infrastructure, implemented at the community or enterprise level that provide collaboration tools, access to services-related **metadata** and thus enable service discovery and use, and technological support for enterprise governance of services. See the Core Enterprise Services [P1175] and related perspectives, especially NCES Directory Services [P1176]. Services are subject to enterprise governance.

> ***Note:*** *Many of the elements of the services-enabling environment participate in the governance structure and processes with participation increasing as the governance matures; however, in this NESI release, such governance of services currently is outside the scope of this perspective.*

DoD leadership has expanded the use of the term "service" beyond mission or business services, as often occurs in some commercial enterprises as well. This is due in part to the fact that the term was in use before the formalized notion of SOA evolved but more so because the benefit of applying principles of service orientation throughout the enterprise architecture enables a degree of uniformity in management of mission and business services plus utilities and infrastructure elements that support and enable them (often called "infrastructure services").

For example, any infrastructure environment utility or function (e.g., a protocol translation function), in good practice, should have defined the party responsible for it, its scope of use and deployment, its interface, rules of access, etc. This data about the utility could be expressed using the same description metadata standard (e.g., **Service Definition Framework** or **SDF**) that is used for a mission service; the utility could be visible and discoverable to the enterprise through the same catalogs and search engines, and a there can be a **Service Level Agreement** (**SLA**) established between the users of the utility and those who are responsible for it. This illustrates the applicability of SOA management approaches to service-enabling utilities and supporting infrastructure elements. The **NCES** enterprise utilities are examples of using the term "service" to describe support environment functions.

The main distinction between an infrastructure and a mission or business service is that an infrastructure service does not represent a primary, distinct mission or business function like a mission or business service does. An infrastructure service is not designed with the flexibility of a mission or business service to be orchestrated into an operational flow or thread. Instead, it might be a part of the underlining infrastructure necessary for mission threads and business processes to execute.

There is a distinction between a service as a service-oriented architecture element (e.g., a service that fetches a specific situational awareness data) and the technology selected to implement it (e.g., a Web service following WS-* specifications, an RSS, etc). Nodes must identify common standards for the modularization, distribution and interaction mechanisms. Service interfaces define the modular boundaries of the provider and consumer. They also serve as the framework for the interactions between provider and consumer **components** and their usage agreements.

Node interaction includes intraNode, interNode and extraNode (the notion that helps understand service interoperability issues). Based on the scope of service use in relation to the Node boundaries and independently of the type of the service (e.g., mission/business or support environment), three groups of services include the following:

Enterprise Service (ES) - a service which has broad applicability/usage across multiple Nodes or across the GIG and typically involves or supports interNode interaction. For services supporting Node operations, loss of an enterprise service can have significant impact on data or process availability necessary for Nodes to operate. An important aspect of Enterprise Services is that their data and interface definitions are collaboratively developed and accepted across the Enterprise but not necessarily centrally governed.

Core Enterprise Service (CES) - a subset of the Enterprise Services where the service is ubiquitous across the Enterprise and, depending on the nature of a CES, the loss of it might have a severe impact on the availability of the necessary data and processes for Nodes and perhaps the GIG to operate. This critical impact potential necessitates that a central coordinating authority act as executive agent for the collaboratively developed and accepted data and interface definitions. The executive agent also probably executes some necessary "core" element of the infrastructure required to support a minimal set of capability in support of the CES.

Local (Internal) Service - a service that typically is mission- or application-specific or provides support to intraNode interaction and operation. This class of service is often designed as a means of distributed application integration; it may be used or reused in other Nodes but the data/interface definition ownership and stewardship responsibilities stay with the original Node, Component or Program.

It is possible that a "community" of Nodes may share services; the threshold at which these services become Enterprise Services is subjective and during that transition, services may have both internal and enterprise characteristics. Services may start out as local and then gather momentum in a community. When the **Community of Interest** (**COI**) advocates standards for that service, it becomes a candidate for an Enterprise Service. ES-track standard services are so critical that the COI identifies an executive agent for coordinating the evolution of the service definition as well as operation of a minimal infrastructure to support interNode and extraNode interactions using that service. Reengineering of services

may be necessary for the services to become suitable for enterprise use (see the Phases of SOA Adoption [P1238] perspective in Part 3: Migration Guidance [P1198]).

The loss of an operationally significant CES or ES does not necessarily imply an impact on a Node's internal operations or its ability to operate independent of the GIG. A local cache, proxy, or alternative source may actually service the request. See the Cross-Domain Interoperation [P1169] and CES and Intermittent Availability [P1168] perspectives for further information.

Access to Core Enterprise Services from Nodes or systems in tactical edge and other environments with either challenged infrastructure performance or extraordinary protection characteristics may also require support for caching, content-filtering, anonymizing, and mediation-proxy interoperability, especially between Core Enterprise Services and the local Node. See the perspectives Design Tenet: Inter-Network Connectivity [P1266], Integration of Legacy Systems [P1135], and CES and Intermittent Availability [P1168] for further information.

Service security is an integral part of securing nodes as well as the infrastructure. Services have two major component families, the "provider" components and the "consumer" components, each managed within the context of its local host. It is essential to harden both properly. Some of the technologies used in this process include but are not limited to: Kerberos, WS-Security, X.509, and **SAML**. See the Integrity [P1334] perspective for more information.

Provider components, such as servers, are often a tightly integrated combination of the local computing infrastructure management, the server host's transport layer port management, and the management model of and infrastructure for the application itself. The use of Web services also requires the management of local Web infrastructure providers.

Consumer components, such as clients and browsers, require computing infrastructure management, user environment management, consumer host transport layer port management, and the standardized end-to-end management of the application itself. Web service components also require management of the local Web infrastructure for consumers.

In addition to the management of the components, service management depends on the scope of the service in question. Some services, especially Network Services [P1353] and Application Layer Protocols [P1355] have such a large impact and their components are so widely distributed that responsibility for management is distributed throughout the enterprise. Such distributed management requires coordination among the providers and is generally standardized in terms of **structured identifier** allocation and assignment as well as synchronization protocols.

Enterprise services, on the other hand, generally have their provider as the primary responsible authority, but due to their wide use also have particular Service Optimization and Scalability [P1327], filtering, aggregation, and federation concerns (See the Utility Services [P1328] perspective for more information). Coordination of distributed management in these cases is often more a matter of federation, mirror-site synchronization and proxy deployment management.

Internal services with a mission focus have a primary responsible authority, the provider, but also require coordination with other partner mission services through orchestration and workflow management techniques and technologies.

One of the challenges in promoting an Internal Service to Enterprise Service is that the service may have to switch from internal, intra-Node infrastructures to standardized, interoperable inter-Node infrastructures. For example, many orchestration technologies require all partner Nodes either have common (shared) or interoperable transport and computing file system infrastructures. Three critical areas for interoperable infrastructures are identifier allocation and assignment, service discovery, and enterprise management monitoring and configuration of components.

Detailed Perspectives

- Core Enterprise Services (CES) [P1175]
- Service Enablers [P1325]
- Service Optimization and Scalability [P1327]
- Utility Services [P1328]

P1175: Core Enterprise Services (CES)

Core Enterprise Services (**CES**) require a centralized governing authority to select, develop and manage the services due to their enterprise-wide scope and importance (see the Services [P1164] perspective). In the DoD, both mandated and organic evolution will define the set of Core Enterprise Services for use across the network. While the exact nature of how CES evolve organically within the DoD is unclear, the DoD Net-Centric Services Strategy (NCSS) [R1313] obligates Nodes to employ a set of DoD Core Enterprise Services that the identified by the DoD ***Enterprise Information Environment Mission Area*** (EIEMA). These services provide a common information environment infrastructure for the purpose of making other services in the enterprise visible and accessible to anticipated and unanticipated users. The CES also enable interoperability across the **Global Information Grid** (**GIG**) and reduce duplication and unnecessary redundancy in the EIEMA portfolio. The EIEMA community will mandate the use of CES across the DoD as the services become available.

Within the DoD, DISA is responsible for defining and developing some of these capabilities through the **Net-Centric Enterprise Services** (**NCES**) program with the following mission:

- Provide executive life cycle management of enterprise capabilities to support the DoD transformation to net-centricity
- Provide executive oversight in planning and delivery of (**ES**) support to mission performance across the Warfighter, Business, and Intelligence Missions Areas
- Provide the infrastructure to publish data/metadata artifacts and enable the DoD Net-Centric Data Strategy

There are four NCES Product Lines [R1259]:

- ***Collaboration*** - Communicate in real-time using voice, text, and video sessions. Supports collaboration between consumers and producers of information to ensure a common understanding and de-confliction of information. For more on Collaboration see the Collaboration Services [P1184] perspective.
- ***Content Discovery and Delivery*** (CD&D) - Enterprise-wide access to shared/stored data; improved situational awareness; ability for user to acquire more information, more quickly, with a smaller footprint. Federated Search is a type of an enterprise Content Discovery Service; for DoD CES implementation see the NCES Federated Search [P1182] perspective.
- ***User Access*** (Portal) - Tailorable user interface providing a window into NCES and access to its capabilities and information.
- ***Service-Oriented Architecture Foundation*** (SOAF) - Loosely-coupled set of services (security, registry, metadata, mediation, etc.) providing foundation for interoperable computing, including the following capabilities that are mapped to services:
  - Enterprise Service Management provides a toolset with a graphic user interface
    - collects standardized metrics for every monitored service through service component management standard interfaces
    - publishes or otherwise makes available collected metrics to authorized and authenticated consumers
    - enables authorized consumers to set behavioral policy thresholds for each metric
    - publishes or otherwise notifies authorized consumers when a metric goes outside a threshold.
    - publishes a catalog of the monitored services and any inter-dependencies and interactions among them, based on a combination of registered and discovered configurations, to authorized consumers
  - Mediation - capabilities for information transformation, service adaptation, and service orchestration (for a discussion about Transformation see the Utility Services [P1328] perspective)
  - Messaging - Messaging provides a federated, distributed, and fault-tolerant enterprise message bus
  - Metadata services - provide the ability for DoD Enterprise systems to discover and manage (publish, make visible, and access) various metadata artifacts critical to a system's and/or a person's ability to exchange and understand data components within the enterprise. They provide visibility of data representations and enable the development and management of data products to support mediation capabilities within the enterprise. The **DoD Metadata Repository** (**MDR**) stores metadata artifacts such as RDBMS schemas, XML schemas, Taxonomies, and XSL

Transforms. The MDR allows categorization of all of the metadata artifacts (and potentially, services, documents, and people) under one or more taxonomies

- People and Service Discovery - See NCES Directory Services [P1176] and Service Discovery [P1181] perspectives.
- Service Security - provides the support necessary to enable DoD net-centricity

For further information on service management, see the ***Management Considerations*** section of the Services [P1164] perspective.

For further information on service security, see the ***Security Considerations*** section of the Services [P1164] perspective.

Detailed Perspectives

- Overarching Issues [P1165]
- NCES Directory Services [P1176]
- Service Discovery [P1181]
- NCES Federated Search [P1182]
- Collaboration Services [P1184]

P1165: Overarching CES Issues

There are particular challenges in implementing and deploying **Core Enterprise Services** (**CES**), especially in a tactical edge environment. Availability of CES will be a continuing challenge until all services reach full maturity and operational status. Designating a CES liaison should help to monitor the availability of CES functionality and report on them back through the engineering processes of the Node and **components** within the Node. Conversely, the engineering processes for the Node should specifically include provisions for incremental implementation of the CES services.

Nodes operating at special classification levels should coordinate with other Nodes within the same level and with DISA to host CES services on the relevant networks.

Overarching Node application Enterprise Services issues include maturity, availability, disconnected operations, cross-domain security, and compliance. These elements equate to the following perspectives:

- Maturity: CES Definitions and Status [P1166]
- Disconnected operations: CES and Intermittent Availability [P1168]
- Cross-domain security: Cross-Domain Interoperation [P1169]
- Compliance: Net-Ready Key Performance Parameter (NR-KPP) [P1170]

Guidance

- G1576: Provide an environment to support the development, build, integration, and test of net-centric capabilities.
- G1626: Identify which **Core Enterprise Services** (**CES**) capabilities the Node **Components** require.
- G1627: Identify the priority of each **Core Enterprise Services** (**CES**) capability the Node **components** require.
- G1629: Identify which **Net-Centric Enterprise Services** (**NCES**) capabilities the Node requires during deployment.
- G1577: Maintain an **Enterprise Service** schedule for interim and final **enterprise** capabilities within the Node.
- G1578: Define a schedule for **Components** that includes the use of the **Enterprise Services** defined within the Node's enterprise service schedule.

Best Practices

- BP1661: Engage with the **Net-Centric Enterprise Services** (**NCES**) program office to explore approaches for mobile use of the **Core Enterprise Services** (**CES**) services in mobile Nodes that rely on **Transmission Control Protocol/Internet Protocol** (**TCP/IP**) for inter-node communication.
- BP1675: In the Node's Web infrastructure, support the technologies and standards used by the **CES** services under development as well as any technologies and standards used for **Community of Interest** (**COI**) services.
- BP1683: Coordinate the Node schedule with the schedules of the **Core Enterprise Service** (**CES**) providers.
- BP1684: Coordinate the Node schedule with the **Component** schedules.
- BP1649: Specifically include provisions for incremental implementation of the **CES** services.
- BP1650: Specifically include provisions for incremental implementation of the hosting Node's **CES** services for Node **Components**.
- BP1695: Designate a **Core Enterprise Services** (**CES**) liaison to monitor the availability of services.
- BP1697: Make the parallel development of **Core Enterprise Services CES** outside the control of the Node a part of the Node's risk management activities.

P1166: CES Definitions and Status

The **Core Enterprise Services** (CES) capabilities are in various states of maturity. Capabilities will be delivered in increments; CES Increment 1 capabilities, shown below, are scheduled for operation beginning in 2008 (source: https://ges.dod.mil/soa.htm; user authorization required).

Service Discovery	Provides a ***yellow pages***, categorized by DOD function, enabling users to advertise and locate capabilities available on the network
Service Security	Provides a layer of defense in depth that enables protection, defense, and integrity of the information environment
Identity Management	Provides the methodology and functions for maintaining information on people, consumers, and service providers. Supports the validation of identity authentication credentials
Service Management	Enables monitoring of DoD **Web services**. Provides reporting of service-level information to potential and current service consumers, program analysts, and program managers
Service Mediation	Allows disparate applications to work together across the **enterprise** by supporting the transformation of information from one format to another, and the correlation and fusion of data from diverse sources. Supports creation and implementation of process workflows across the enterprise
Machine-to-Machine Messaging	Provides reliable **machine-to-machine** message exchange across the enterprise
Metadata Services	Provides access to **Extensible Markup Language** (XML) data elements, taxonomy galleries, schemas, and validation and generation tools for DOD software developers
DoD Web Services Profile	Provides specifications and implementation guidelines to maximize interoperability across DOD Web service implementations

NCES Increments will be rolled out every 24-26 months. Consider the NCES increment schedule in scheduling Node evolution in coordination with systems within the Node.

Guidance

- G1301: Practice layered security.

P1168: CES and Intermittent Availability

Core Enterprise Services (CES) may be unavailable for several reasons, including loss of connectivity, actual service unavailability, or service rejection. There are two related challenges: how to handle lapses in the availability of CES services and how to align inter-Node and intra-Node solutions. The lack of availability of CES services must not disrupt intra-node availability of locally hosted services. While alignment of intra- and inter-node technical solutions is very desirable, the interface to locally hosted **Components** must not be dependent on the availability of CES services.

Specific guidance is largely dependent upon the specific Node operating environment and mission. There are some basic options for meeting these challenges:

- Locally host failover copies of certain CES services. Components that are dependent upon **Enterprise Services** for infrastructure functions, such as security, continue to operate after failing over to the local instances until **enterprise** accessibility is re-established. This approach requires replication of enterprise services data (the data used by the enterprise services) between the local failover services and the "master" enterprise services. It also requires development of failover behavior in the applications, services, and infrastructure.
- Develop Components to be adaptive, applying default rules and behaviors when Enterprise Services are inaccessible. This approach, along with the definition of the default rules and behaviors would depend on factors such as the sensitivity and importance of the information involved. For example, access control decisions might default to local capabilities such as **Active Directory** local user accounts. Or local caching might be used to retain the most recently known values for information such as previously discovered services.
- Employ separate external-facing and internal-facing implementations of published services so that external disruptions do not affect local accessibility. The external-facing copy of the service could use Enterprise Services, and the internal-facing copy could implement local Node behavior. As an example, the external-facing copy could implement **Public Key Infrastructure** (PKI) **authentication** and **authorization**, whereas the internal-facing copy could implement Active Directory security. The challenge in this approach is in the coordination of the external-facing and internal-facing copies of such services, such as to provide shared access to databases or replication of data between the external-facing and internal-facing implementations.

Nodes and Components will likely employ some combination of, or evolution of, the above options.

Uniformity and alignment between the technical mechanisms for accessing local services and Enterprise Services should be an objective. Where possible, the burden of providing such uniformity and alignment should rest on the Node infrastructure, rather than the individual Components within the Node, thus isolating the complexities and making them more manageable. Consider the necessity of using CES-provided **Software Developers Kits** (**SDKs**) and **Key Interface Profile** (KIP) compliance when formulating an approach; use of an approved SDK may drive separation of external-facing and internal-facing implementation described in the last option above. Finally, the immaturity of the CES services and the alignment of local and external services access, as a whole, should figure prominently in the risk management activities of the Node and Components within the Node.

Guidance

- G1630: Comply with the applicable **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) for implemented **Core Enterprise Services** (**CES**) in the Node.
- G1631: Expose **Core Enterprise Services** (**CES**) that comply with the applicable **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in all Node services **proxies**.

Best Practices

- BP1651: Ensure **Node Components** have access to **Core Enterprise Services**.

P1169: Cross-Domain Interoperation

By and large, the implementation of net-centric concepts across security domains has not been defined. Trusted guards do not act as network **routers**; information to be transferred across a guard is delivered to the guard, processed, and then delivered to a defined endpoint on the other side if the rules are satisfied. The guard in the middle disrupts the normal pattern for use of the **CES** services.

In order for **services** to work through the trusted guards that interconnect different domains, there must be a well defined set of messages that can be passed through the guard to effect the conversation necessary to use the service and return results. This restriction, if built into the service's interface, could be unduly restrictive on the design of the interface.

It may be more practical for each such service to provide service proxies for use in the other security domains, and corresponding client proxies in the local domain. The server **proxy** and client proxy for the service might then communicate across the trusted guard in a private, high efficiency manner that the guard can process. But even this approach is restrictive in that the server proxies have to be installed in the other security domains, and this departs from some fundamentals of net-centric concepts such as dynamic **service discovery**.

Until such approaches are prototyped and explored more fully, Nodes should anticipate that services will not be capable of cross-domain invocation. Furthermore, for services that have utility in other security domains, implementer should consider providing copies of such services for hosting in the other domains, and use **XML** document transfers across the trusted guard to keep the copies in synchronization. This approach depends on many factors, and may not be suitable for all services.

Guidance

- G1613: Prepare a **Node** to host new **Component services** developed by other Nodes or by the **enterprise** itself.

Best Practices

- BP1691: Use **Node** implemented **Service Discovery** (**SD**) to meet compartmentalization needs.
- BP1698: Plan for the event that **Component** services within a **Node** cannot be invoked across security domains.
- BP1614: Plan a contingency response to the **Node** becoming a new **component service** within another Node.

P1170: Net-Ready Key Performance Parameter (NR-KPP)

The **Net-Ready Key Performance Parameter** (**NR-KPP**) provides a means to assess net-ready attributes required for both the technical exchange of information and the end-to-end operational effectiveness of that exchange. The NR-KPP replaces the Interoperability KPP, and incorporates net-centric concepts for achieving **Information Technology** (**IT**) and **National Security Systems** (**NSS**) interoperability and supportability. The NR-KPP assists Program Managers, the test community, and Milestone Decision Authorities in assessing and evaluating IT and NSS interoperability.

The NR-KPP assesses information needs, information timeliness, **information assurance** (**IA**), and net-ready attributes required for both the technical exchange of information and the end-to-end operational effectiveness of that exchange. The NR-KPP consists of verifiable performance measures and associated metrics required to evaluate the timely, accurate, and complete exchange and use of information to satisfy information needs for a given capability. Program managers will use the NR-KPP documented in **Capability Development Documents** (**CDD**) and **Capability Production Documents** (**CPD**) to analyze, identify, and describe IT and NSS interoperability needs in the **Information Support Plan** (**ISP**) and in the test strategies in the **Test and Evaluation Master Plan** (**TEMP**).

The Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01E, *Interoperability and Supportability of Information Technology and National Security Systems*, 15 December 2008, [R1175] removed the **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**), integrating the components of the former NCOW RM into other elements of the NR-KPP. The following five elements now comprise the NR-KPP:

- Compliant solution architecture
- Compliance with DOD Net-Centric Data and Services strategies ([R1172] and [R1313],respectively), including data and services exposure criteria
- Compliance with applicable GIG Technical Direction to include **DISR**-mandated IT Standards reflected in the **TV**-1 and implementation guidance of GIG Enterprise Service Profiles (GESPs) necessary to meet all operational requirements specified in the DoD Information Enterprise Architecture and solution architecture system/service views
- Verification of compliance with DOD IA requirements
- Compliance with supportability elements to include, spectrum analysis, Selective Availability Anti-Spoofing Module (SAASM) and the Joint Tactical Radio System (JTRS)

Detailed Perspectives

- Information Assurance (IA) [P1171]
- Net-Centric Operations and Warfare Reference Model (NCOW RM) [P1172]
- Key Interface Profile (KIP) [P1173]
- Integrated Architectures [P1174]

P1171: Information Assurance (IA)

Most Nodes, when delivering a capability to the warfighter or business domains, will use **Information Technology** (IT) to enable or deliver that capability. For those Nodes, developing a comprehensive and effective approach to **IA** is a fundamental requirement and is key in successfully achieving Node's objectives. The DoD defines IA as follows [see DoDD 8500.1, Enclosure 2 Definitions (E2.1.17)]:

Information Assurance (IA). Measures that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. This includes providing for the restoration of information systems by incorporating protection, detection, and reaction capabilities.

DoD policy and implementing instructions on information assurance are in the 8500 series of DoD publications. Nodes and **Components** for programs should be familiar with statutory and regulatory requirements governing information assurance and understand the major tasks involved in developing an IA organization, defining IA requirements, incorporating IA in the Node's and Component architecture, developing an acquisition IA strategy (when required), conducting appropriate IA testing, and achieving IA certification and accreditation for the program.

Guidance

- G1632: Certify and accredit Nodes with all applicable DoD **Information Assurance** (IA) processes.
- G1633: Host only DoD **Information Assurance** (IA) certified and accredited **Components**.
- G1634: Certify and accredit **Components** with all applicable DoD **Information Assurance** (IA) processes.

P1172: Net-Centric Operations and Warfare Reference Model (NCOW RM)

The **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**) represented strategies for transforming the **enterprise** information environment of the Department of Defense. It was an architecture-based description of activities, services, technologies, and concepts to enable a net-centric enterprise information environment for warfighting, business, and management operations throughout the DoD. Included in this description were activities and services required to establish, use, operate, and manage this net-centric enterprise information environment. Major activity blocks included the generic user-interface, the intelligent-assistant capabilities, the net-centric service (core, **Community of Interest**, and enterprise control) capabilities, the dynamically allocated communications, computing, and **storage** media resources, and the enterprise information environment management components. Also included was a description of a selected set of key standards and/or emerging technologies that would be needed as the NCOW capabilities of the **Global Information Grid** (**GIG**) were realized.

Transforming to a net-centric environment requires achieving four key attributes: reach, richness, agility, and assurance. The initial elements for achieving these attributes include the *DoDNet-Centric Services Strategy*[R1313], the *DoD Net-Centric Data Strategy*[R1172], and the *DoD Information Assurance (IA) Strategy*[R1345] to share information and capabilities. The NCOW RM incorporated these strategies as well as net-centric results produced by the Department's **Horizontal Fusion** (**HF**) pilot portfolio.
The NCOW RM provided the means and mechanisms for acquisition program managers to describe their transition from the current environment (described in ***GIG Architecture Version 1***) to the future environment (described in ***GIG Architecture Version 2***). In addition, the NCOW RM was a key tool during program oversight reviews for examining integrated architectures to determine the degree of net-centricity a program possessed and the degree to which a program could evolve to increased net-centricity. Compliance with the NCOW RM was one of the four elements that initially comprised the **Net-Ready Key Performance Parameter** (**NR-KPP**).

> ***Note:*** *The NCOW RM was a key compliance mechanism for evaluating DoD information technology capabilities and the NR-KPP in accordance with Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01D, Interoperability and Supportability of Information Technology and National Security Systems, 8 March 2006. The 15 December 2008 revision to this instruction, CJCSI 6212.01E, removed the NCOW RM element of the NR-KPP, integrating the components of the former NCOW RM into other elements of the NR-KPP.*

Guidance

- G1636: Comply with the **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**).

P1173: Key Interface Profile (KIP)

Note: *Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01E[R1175], revised 15 December 2008, deletes the* ***Key Interface Profile*** *(****KIP****) element of the NR-KPP and replaces it with the "Technical Standards/ Interfaces" element. This revision further indicates that* ***Global Information Grid*** *(****GIG****) Enterprise Service Profiles (GESPs) are evolving to provide a net-centric oriented approach for managing interoperabilty across the GIG based on the definition and configuration control of key interfaces and enterprise services. The* ***Defense Acquisition University*** *(****DAU****) Interim Defense Acquisition Guidebook, Chapter 7, contains additional information.*

The following information is from an earlier version of the *Defense Acquisition Guidebook* (specifically, Chapter 7.3.4.2). A KIP is the set of documentation produced as a result of interface analysis which designates an interface as key; analyzes it to understand its architectural, interoperability, test and configuration management characteristics; and documents those characteristics in conjunction with solution sets for issues identified during the analysis. The profile consists of refined operational and systems view products, Interface Control Document/Specifications, Systems Engineering Plan, Configuration Management Plan, **Technical Standards View** (TV-1) with SV-TV Bridge, and procedures for standards conformance and interoperability testing. Relevant GIG KIPs, for a given capability, are documented in the **Capability Development Document** and **Capability Production Document**. Compliance with identified GIG KIPs are analyzed during the development of the **Information Support Plan** (**ISP**) and **Test and Evaluation Master Plan**, and assessed during **Defense Information Systems Agency Joint Interoperability Test Command** (**JITC**) joint interoperability certification testing. An interface is designated as a key interface when one or more the following criteria are met:

- The interface spans organizational boundaries.
- The interface is mission critical.
- The interface is difficult or complex to manage.
- There are capability, interoperability, or efficiency issues associated with the interface.
- The interface impacts multiple acquisition programs.

Program manager compliance with applicable GIG KIPs is demonstrated through inspection of **Joint Capabilities Integration and Development System** (**JCIDS**) documentation and test plans, and during JITC interoperability certification testing (see CJCSgttp://kips.disa.mill 3170.01 and CJCSI 6212.01 for detailed discussions of the process).

Guidance

- G1630: Comply with the applicable **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) for implemented **Core Enterprise Services** (**CES**) in the Node.
- G1631: Expose **Core Enterprise Services** (**CES**) that comply with the applicable **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in all Node services **proxies**.

Best Practices

- BP1685: For **Key Interface Profile** (**KIP**) specifications that are not available or insufficiently mature, implement a "best effort" by following the published intent of functionality and monitor or participate in the relevant specification development body.

P1174: Integrated Architectures

The **DoD Architecture Framework** (**DoDAF**), available via the ***General Public Documents*** Quick Link on the DoD Architecture Registry System Welcome Page, provides the rules, guidance, and product descriptions for developing and presenting architecture descriptions to ensure a common denominator for understanding, comparing, and integrating architectures. An integrated architecture consists of multiple views or perspectives (**Operational View** [**OV**], **Systems and Services View** [**SV**], **Technical Standards View** [**TV**] and **All-Views** [**AV**]) that facilitate integration and promote interoperability across capabilities and among related integrated architectures.

- The OV is a description of the tasks and activities, operational elements, and information exchanges required to accomplish DoD missions.
- The SV is a description, including graphics, of systems and interconnections providing for, or supporting, DoD functions.
- The TV is the minimal set of rules governing the arrangement, interaction, and interdependence of system parts or elements, whose purpose is to ensure that a conformant system satisfies a specified set of requirements. Technical Views include approved standards from the **DoD Information Technology Standards Registry** (**DISR**).[R1179]
- The AV products provide information pertinent to the entire architecture but do not represent a distinct view of the architecture. AV products set the scope and context of the architecture.

The **Global Information Grid** (**GIG**) architecture describes the basic, high level architecture in which Nodes reside. It is an integrated architecture consisting of the various DoDAF views. It provides a common lexicon and defines a basic infrastructure for the performance of information exchanges with other GIG Nodes using the **GIG Enterprise Services** (**GES**) that DISA is developing as part of the **Net-Centric Enterprise Services** (**NCES**) program.

Guidance

- G1635: Make Nodes that will be part of the **Global Information Grid** (**GIG**) consistent with the *GIG Integrated Architecture*.

P1176: NCES Directory Services

Secure inter-node interoperability relies heavily on the ability to lookup information about people and objects or devices across the breadth of the **Global Information Grid** (**GIG**). The technologies that support this form of discovery are known collectively as directory services. There are several standardized and layered directory services. The lower layer directory services primarily discover Internet Hosts on which data, applications, services and people's accounts reside.

The best known of the lower layer directory services is the **Domain Name System** (**DNS**). The lower layer directory services also include various host identification services such as the **Dynamic Host Configuration Protocol** (**DHCP**). The Network Services [P1353] perspective covers these services in more detail. More localized enterprise directory services include Windows directory services (such as Windows Internet Name Service or WINS) and Novell Directory Services (NDS). These services are confined within the local area network or virtual local overlay network and require the **Net-Centric Enterprise Services** (**NCES**) directory services to interoperate beyond the Node or its local infrastructure.

For performance and scalability reasons, core lower layer directories usually are constrained to critical services such as **Public Key Infrastructure** (**PKI**) support for email and people (such as administrative user email accounts) in addition to their primary function as a host identity registry.

The NCES service taxonomy includes NCES Directory Services under the scope of CES People Discovery as part of Service-Oriented Architecture Foundation product line (see [R1259]). NCES People Discovery provides services to publish and find, via LDAP-standard interfaces, available information on GIG users and connected devices. The Joint Enterprise Directory Services (JEDS) provides user information aggregated from a number of DoD repositories.

Nodes routinely use directory services today, such as Microsoft **Active Directory** and the DoD PKI Global Directory Service (GDS). Although implementations are widespread across the GIG, there is limited coordination and synchronization, creating pockets of information that must be unified. There are also substantial differences among implementations, including naming conventions. This situation is made more complex by the fact that these directories are typically also integral to a Node's security and system administration, supporting such basic functions as user login.

SOA Directory Services

A SOA-specific registry and directory service is **Universal Description Discovery and Integration** (**UDDI**). See the Service Discovery [P1181] perspective for detailed information.

Guidance

- G1625: Provide a **commercial off-the-shelf** Directory Service that all of the **components** of a Node can use.
- G1637: Make Node-implemented **directory services** comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**).
- G1638: Comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node directory services **proxies**.

Best Practices

- BP1686: Align Node interfaces to **Components** for directory services with the guidance being provided by the Joint Directory Services Working Group (JDSWG) and sub-working groups, including such guidance as naming conventions, federation, and synchronization.
- BP1687: Follow **Active Directory** naming conventions defined in the *Active Directory User Object Attributes Specification* as required by the DoD **CIO** memorandum titled *Microsoft Active Directory (AD) Services*.

P1181: Service Discovery

The ability to discover services is a major factor in the enablement of using and sharing services in the enterprise. The discovery concept relies on human- and machine-usable registries for maintaining metadata descriptions of information and services. The intent of these "service registries" is to provide all of the information required for an application developer to locate and use an appropriate service; for example, determine the features and functions the service provides, identify how to invoke the service, discover the supported **Quality of Service** (**QoS**), understand how to contact the service owners, and determine where the service resides. In the case of highly mature services (see the set of Migration Patterns [P1201] perspectives for SOA maturity discussions), Nodes and Components should also be able to discover dynamically where Component services and information reside in the **Global Information Grid** (**GIG**) and bind to those providers at runtime.

The DISA **Net-Centric Enterprise Services** (**NCES**) program provides such a registry/repository as part of the NCES SOA Foundation product line. NCES Service Discovery consists of a **commercial off-the-shelf** (**COTS**) **Universal Description, Discovery, and Integration** (**UDDI**) registry customized to provide service governance as well as enhanced end user access. Web services are also available to enable service publishing and service discovery at the application layer.

Nodes face several implementation choices regarding the alignment of **Component** and Node approaches to service discovery. Register Components that the Node exposes with the DISA-hosted registries so that the Component services are visible to other Nodes. Internal-facing services that are not likely to be of value beyond the boundary of a Node do not have to be discoverable, although it is a good practice. Implementing service discovery within a Node can support availability of Component services within the Node.

Guidance

- G1639: Describe **Components** exposed by the Node as specified by the **Service Definition Framework**
- G1640: Register **Components** exposed by the Node with the **DISA**-hosted registries.
- G1641: Comply with the Service Discovery **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node-implemented **Service Discovery** (**SD**).
- G1642: Comply with the **Service Discovery Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node **Service Discovery** (**SD**) **proxies**.

Best Practices

- BP1690: Use Node implemented **Service Discovery** (**SD**) for high availability.
- BP1691: Use **Node** implemented **Service Discovery** (**SD**) to meet compartmentalization needs.

P1182: NCES Federated Search

The DISA **Net-Centric Enterprise Services** (**NCES**) program description of Content Discovery states that Content Discovery provides a standard, vendor neutral approach for exposing metadata to the **Global Information Grid** (**GIG**). It consists of three components:

- ***Centralized Search*** - Web content crawled by Intelink
- ***Federated Search*** - Interface for submitting search queries and returning aggregated results
- ***Enterprise Catalog*** - Interface for information producers to update enterprise metadata catalogs

The capability allows searching across a set of Content Discovery Services and yielding an integrated result. The Federated Search service allows sending a query to a large set of disparate data providers, collecting the results generated by each, and presenting the results back to the user after de-duplicating, ranking, etc. This allows a user to submit a query from one place using one syntax and retrieve relevant data from many sources across DoD. This approach leverages existing data sources and production processes.

Federated Search implementation is a set of cooperating Web services. These services talk to each other using a common specification. The specification defines the communication of the query and the results from the query. It describes not only the meaning, but also the format of the data that services exchange.

The Federated Search service uses the **Defense Discovery Metadata Specification** (**DDMS**) to represent the concepts of a query as well as the resource result records, called meta cards, that a search result generates. Data providers match outgoing queries against the resource meta cards to generate search results. The DDMS ties the queries to the results using a common vocabulary.

The domain of the Federated Search service is limited to the provider sites the sponsoring organizations make available for the DoD enterprise. The Federated Search service does not provide visibility or access to private provider sites that do not participate in the Federated Search service. Each **Node** should implement Federated Search - **Registration Web Service** (**RWS**) and **Search Web Service** (**SWS**). Data producers use the RWS to register content sources; the SWS is searches for content from the registered sources.

Guidance

- G1643: Comply with the **Federated Search** - **Registration Web Service** (**RWS**) **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node implemented Federated Search - Registration Web Service (RWS).
- G1644: Comply with the **Federated Search** - **Search Web Service** (**SWS**) **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) in Node implemented Federated Search - Search Web Service (SWS).
- G1645: Implement a local **Content Discovery Service** (**CDS**).
- G1646: Comply with the directory services **Global Information Grid** (**GIG**) **Key Interface Profiles (KIPs**) in Node **Federated Search** Services **proxies**.
- G1647: Provide access to the **Federated Search** Services.

Best Practices

- BP1648: Host the **Registration Web Service** (**RWS**) registration **portlet** in the Node.
- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.

P1184: Collaboration Services

Collaboration tools provide a virtual meeting room environment for human interaction. The virtual environment enables multimedia collaboration (text, voice, and video) in multiple modes (person-to-person, open chat, restricted meeting, etc.) and application broadcasting and sharing.

A 2 February 2009 DoD **CIO** memo, *DoD Enterprise Services Designation*, describes the designated DoD **Enterprise Services**, including collaboration services. The **DISA Joint Interoperability Test Command** (**JITC**) has validated a suite of collaboration tools and standards called the **Defense Collaboration Tool Suite** (**DCTS**) for interoperability and operational use. The DCTS **Collaboration Management Office** (**CMO**) within DISA is responsible for fielding, sustaining, and managing the life cycle of DCTS. Collaboration products approved for interoperability are listed at http://jitc.fhu.disa.mil/washops/jtcd/dcts/status.html. Products certified for use on the **Secret Internet Protocol Router Network** (**SIPRNet**) are listed at http://jitc.fhu.disa.mil/washops/jtcd/dcts/projects.html.

Programs are not to implement chat services or renew licenses on existing services that overlap with approved DoD Enterprise Services without a waiver. Circumstances that may justify a waiver include challenging or hostile operational environments that have additional performance, including **quality of service** (**QoS**), requirements that the designated DoD Enterprise Services cannot adequately meet. If a program utilizes a locally developed or provided chat service, the NESI Text Conferencing [P1388] perspective provides applicable reference information and guidance. Any such locally developed or provided service should conform with standards registered within **Defense IT Standards Registry** (**DISR**), applicable **security technical implementation guides**, and products from JITC list.

Detailed Perspective

- Text Conferencing [P1388]

Best Practices

- BP1692: Determine which Collaboration Service vendor offering to employ in a disadvantaged environment or separate network.
- BP1693: Make sure that **collaboration** products used to satisfy urgent requirements are from the **JTIC** list.

P1325: Service Enablers

The following basic factors enable service use:

- service is identified by standard **structured identifier** such as a **Uniform Resource Identifier** (**URI**)
- service is advertised across the enterprise
- service is discoverable across the enterprise

In addition to these basic factors, give careful consideration to the following separate but related topics:

- ***Service Provider*** - service deployment, provisioning, service consumer relationship maintenance, change management
- ***Service Consumer*** - service selection, integration and interoperability, service provider relationship maintenance, change management
- ***Service Infrastructure*** - service advertisement and discovery scope management, isolation, aggregation, mirrors and proxies, capacity and mission assurance management, etc.

For interaction (including interNodal and extraNodal) with the **Global Information Grid** (**GIG**), the DISA **Net-Centric Enterprise Services** (**NCES**) program provides a **Core Enterprise Services** (**CES**) level implementation for some of these enablers (e.g., Discovery Services).

Service Management interoperability depends on management standards such as those from the Information Technology Infrastructure Library (ITIL), the Distributed Management Task Force (DMTF), the **International Telecommunications Union** (**ITU**) and the Telemanagement Forum's extended Telecommunications Operations Map (model).

> ***Note:*** *In the case of a composite service, register each of the services that comprise it and provide each service's own unique URI and description.*

Service Identification

URIs uniquely identify **HTTP**-based services, and their identifiers are managed in accordance with Command Structure, Doctrine and Commander's Intent.

Service Publication and Advertisement

Provide enough semantic information in service advertisements to allow perspective service consumers to determine whether the service is suitable for a particular application. The service consumer should not have to examine the service code to make this determination.

Each service provider registers and provides a public abstract interface of its services and data to include its transport and information assurance bindings.

For further information see the Service Discovery [P1181], Service Definition Framework [P1296], and Universal Description, Discovery, and Integration (UDDI) [P1075] perspectives.

Service Discovery

A service may be discoverable a number of ways: by searching a repository such as the **DoD Metadata Registry**, by searching a well-known service catalog technology such as multi-cast catalog or anycast catalog or by searching a **UDDI** directory service, or by using a generic search engine such as Google.

Detailed Perspectives

- Service Discovery [P1181]
- Information Exchange Patterns [P1326]

P1326: Information Exchange Patterns

Three fundamental information exchange patterns prevalent in DoD enterprise are request/response, publish/subscribe and streaming media. Different **Service Level Agreements** (**SLA**) and **Quality of Service** (**QoS**) requirements, especially in the area of transport infrastructure, distinguish these usage patterns. Consequently, they are sensitive to deployment at the Tactical Edge.

Request / Response

While considered a "classic" in client-server architectures, the request/response messaging exchange pattern is also fundamental to the **Service-Oriented Architecture** (**SOA**) style. A service Consumer sends a request message to a service Producer. The Producer processes the message and executes appropriate service operations based on the content of the message. Following the completion of these operations, a response message is returned to the Consumer. This response message may return the requested information or notification of an operation complete (or an exception).

While this pattern is typically implemented in a purely synchronous fashion (as in **Web service** calls over **HTTP**, where the requester holds a connection open and waits until the response is delivered or the timeout period expires), asynchronous implementations of the request/response pattern are also valid.

Publish / Subscribe

Publish/subscribe is a message exchange pattern in which clients address messages to a specific node in a content hierarchy, called a topic. Publishers and subscribers are generally anonymous and can publish or subscribe dynamically to the content hierarchy. The system takes care of distributing the messages arriving from a node's multiple publishers to its multiple subscribers.

This pattern usually is used to distribute events (e.g., notifications about changes in shared state in the architecture) to multiple interested parties as soon as the events become available. An event contains enough information for the subscriber to allow it to initiate an appropriate action, which could include invoking a service. For example, a service consumer interested in a particular remote data subscribes to RSS notifications about changes in or about that data (e.g., a change in data location). When the notification is received, the consumer requests a Web service using parameters provided in the notification and obtains the update. The event itself could be a result of the execution of a service or a result of processing of one or more other events.

This pattern typically is implemented in a loosely coupled asynchronous fashion. One of the main reasons for this is that at the time of the event the networking link with the consumer might be unavailable or the consumer could be down. This requires an intermediary in the form of a queue or other type of agent to store the event message until consumer is able to receive and process it. The degree of message persistence (and therefore the robustness of the system) varies among implementations.

For further information on this topic see the Processes [P1342] perspective.

Streaming and Isochronous Flows

There is a class of data flows such that the flow can be processed as a steady and continuous stream. Noted for their Quality of Service requirements, particularly their sensitivity to variance in inter-packet delay, this class of data includes voice, video and interactive services such as remote control and collaboration.

P1327: Service Optimization and Scalability

Optimization and scalability techniques generally improve application performance by increasing throughput and decreasing latency. Many tactical edge environments are characterized by low-bandwidth and intermittent communications, as well as other resource shortfalls. Optimization and scalability services make the best of challenged resources.

The subsections below describe several representative optimization/scalability techniques; many additional pertinent optimization/scalability techniques exist. Further, there are many varieties of each optimization/scalability technique in commercial industry as well as purpose-built renditions for the military domain, so definitions may vary among vendors.

Caches and compression are common technological threads in performance optimization. Caches are local temporary storage areas for when rapid or frequent access to data or objects is necessary, but they do not transform the data proper. Compression reduces the amount of data in a sequence of bits or bytes for concise transmission and then reconstructs it for access.

Caching

Caching is local storage of remote data designed to reduce unnecessary transfer of data. Caching may improve throughput and decreases latency by avoiding unnecessary trips across the network.

Object caching is very different than byte caching in that it is often protocol/application specific and is an all-or-nothing affair. If the cache contains the object, the user gets access to the object from a local store extremely quickly. Object caching can greatly reduce, almost to zero, the bandwidth and the latency of Web applications. The only transactions that cross the wide area network (WAN) are a quick check to ensure that the copy in cache is still current.

A typical design of application servers includes pools and caches of the internal container services objects that allow the architect to tune the server resources according to the application specifications for performance, scalability, and availability.

Compression

The goal of data compression is to represent an information source (e.g., a data file, a speech signal, an image, or a video signal) as accurately as possible using the fewest number of bits. Data compression is particularly useful in communications because it enables devices to transmit the same amount of data in fewer bits. There are a variety of data compression techniques, but only a few have been standardized.

The **International Telecommunications Union** (**ITU**) has defined a standard data compression technique for transmitting faxes (Group 3 standard) and a compression standard for data communications through modems (V.42bis). In addition, there are file compression formats, such as ARC and ZIP. Backup utilities, spreadsheet applications, and database management systems also use data compression. Certain types of data, such as bit-mapped graphics, can be compressed to a small fraction of their normal size.

Byte caching (sometimes referred to as dictionary or delta-based compression) is a combination technique that relies on a low-level cache of small, sub-application-object pieces of information to detect compressible, repetitive patterns in application cache traffic. It then symbolizes those patterns with a token, and sends the token in lieu of the bulky traffic; tokens typically are a byte or two and symbolize large blocks (e.g., 64KB). The cache on the far end matches the token with the original block of data, reconstitutes the traffic, and sends it on to the application or user (whichever is appropriate).

Protocol Optimization

Protocol optimization aims to reduce latency by removing inefficiencies in key protocols. For example, **TCP** and **HTTP** protocol optimization make Web traffic more efficient over the WAN by removing the unnecessary roundtrips that the protocols introduce as part of their set-up processes.

Load Balancing

Load balancing is a technique (usually performed by load balancers) to spread work among two or more computers, network links, central processing units (CPUs), hard drives, or other resources, in order to get optimal resource utilization, throughput, or response time. These tunable pools of infrastructure resources are managed by a combination of resource capacity metrics and load-balancing algorithm.

Typical industry standard load balancing algorithms available today include the following:

- Round Robin
- Least Connections
- Fastest Response Time
- Weighted Round Robin
- Weighted Least Connections
- Custom rating values assigned to individual servers in a pool, for example server ratings based on delay measurements provided by SNMP or other communication mechanism

Application Server Offload

Application server offload services scale applications by offloading processing tasks from the application servers to purpose built hardware and software devices. For example, compression computations consume CPU resources on servers. Many vendors offload those computations onto purpose-built hardware that performs compression at wire speeds.

P1328: Utility Services

Services use various common filtering, aggregation and data transformation techniques. The techniques in the following subsections are not an exhaustive set but they are of particular use for environments with constrained resources such as the tactical edge.

Smart Content Filtering

Smart filtering and aggregation services, in conjunction with **Quality of Service** (**QoS**) mechanisms, are needed at key information distribution nodes, such as airborne command and control (**C2**) centers (e.g., AWACS) at the tactical edge, to effectively and efficiently distribute information across the wide area network (WAN) and to/from end users on a priority basis.

Smart filtering services enable fine grain filtering based on the full content of each message. With such pinpoint filtering, users may receive just the information that they request (as long as they are authorized,) which minimizes bandwidth utilization. If smart filtering is coupled with QoS mechanisms, then the user will be able to receive just the information subscribed to on a priority basis.

Purpose-built content/message routers can provide full content monitoring and filtering on a per user and per application basis with real-time performance.

Content Aggregation

There are points in the network where information naturally aggregates as it moves towards its destination. For example, information from a squad of solders may flow through the vehicle's communication system. Further, information from a number of vehicles may flow through a battlefield node that intentionally is provisioned to have higher bandwidth and more reliable connectivity than other nodes. User generated packets are introduced to the network and move through the aggregation points, where information aggregation services are applied.

An example of an information aggregation service follows:

Rules in the aggregation point's router ingress interface identify the packets based on network service, protocol, destination, or some other unique factor. The router forwards the packets to a local application that places them into queue for that particular type of information. Periodically, with time intervals perhaps measured in 10s of seconds as dictated by mission need, the application takes the queue contents and builds an outbound packet. The constructed packet payload is the contents of the queue. It is then forwarded towards the destination using an appropriate transport protocol for the intended operational environment.

Transformation

Transformation includes translation between transport mechanisms or data formats as well as protocol mediation. Examples include the following:

- Conversion between two different message formats, such as two tactical data links (e.g., Link 16 and Variable Message Format or VMF)
- Conversion between two XML data formats

Standards such as **XSLT** enable transforming the XML content from one provider to another XML data mode that another consumer can use. The NCES Adapter Library translates information formats from popular standards to XML and translates from XML to other popular information format adapters (provided by the NCES Mediation Services product line). For more detail see the XSLT [P1106] perspective.

Compression

Compression has important applications in the areas of data transmission and data storage. The number of applications processing large volumes of data is increasing, while the proliferation of communication networks is resulting in greater transfer of data over communication links. Compressing data, both during transmission and while at rest, often leads to reduced costs associated with data transportation and storage.

Reducing the amount of data transmitted has the effect of increasing the capacity of the communication channel. This additional capacity may be used to transport additional data or in some cases allow for reduced queuing time for more critically important messages. The additional capacity also allows for additional error detection and/or correction data which increases robustness and reliability of the communication channel.

Similarly, compressing a file to half of its original size is equivalent to doubling the capacity of the storage medium. It may then become feasible to store the data at a higher, thus faster, level of the storage and reduce the load on the input/output channels of the computer system. The more that storage space is conserved, the more storage is available for other uses.

There are various algorithms for data compression. While, in principle, it is possible to use any general purpose compression algorithm on any type of data, many are unable to achieve significant compression on data that is not of the form for which they were designed to compress. The ability to compress depends on the inherent redundancy in the information to be compressed.

Compression algorithms fall into two categories, ***lossy*** and ***lossless***. Lossy algorithms reduce the size of the data through compression but lose fidelity in the process (often with the trade-off of increased compression of the data). On the other hand, lossless algorithms reduce the size of the data through compression techniques that result in no loss of fidelity or accuracy of the data. In other words, lossless algorithms allow for exact recreation of the data to its state before compression. Both categories of data compression are useful depending on the given requirements.

The selection of an appropriate compression algorithm for a given application depends on a number of parameters including redundancy within the data, noise within the data, tolerance to the loss of fine detail, available bandwidth, storage capacity, and the speed of the compression and decompression processes. Shannon's Theorem and subsequent algorithm standards relate all these factors; Shannon's Theorem also sets theoretical bounds on the possible compression available without introducing errors which would distort the content.

For example, a binary string of ones and zeros is generally not compressible unless there are long strings of repeated ones or zeros imbedded in it. Given simple redundancy at the bit level, run length encoding, which replaces the string by the symbol and the number of repeats, is possible. Alphabetic text in a human language has slightly more complicated redundancy and a lossless technique called Huffman coding is preferred. There are likewise specialized algorithms for video, audio, and graphics such as used in the following standards MPEG-2, Ogg Vorbis, and JPEG.

Best Practices

- BP1711: Use the **CES** Mediation Service, or a locally hosted copy, when **XML** document translation between **schemas** is a necessity.
- BP1712: Register developed mappings in the **DoD Metadata Registry**.

P1369: Internationalization Services

This service area relates to the standards for the internationalization of applications. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- Data Modeling [P1003]
- Designing User Interfaces for Internationalization [P1112]

P1112: Designing User Interfaces for Internationalization

Internationalization is the process of generalizing software so that it is interoperable with multiple languages (i.e., locales) and cultural conventions without the need for re-design or re-compilation. If an application designed for a U.S. audience will be used in combined or coalition warfare operations, it needs to provide a user interface that matches users' expectations, interacts with users in their native language, and displays data in a manner that is consistent with users' cultural conventions. The purpose of this perspective is to provide a starting reference for developers needing to support internationalization and provides best practices and resources.

Best Practices

- BP1764: Make all localizable user interface elements such as text and graphics externally configurable.
- BP1765: Declare the encoding type for all user interface content.
- BP1766: Develop user interfaces to accommodate variable syntactic structure for messages.

P1370: Operating System Services

This service area relates to operating system services between an application and the computing platform. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- Software Communications Architecture [P1087]
- Software Security [P1065]

P1065: Software Security

Security is a top priority in the nation's agenda. It is more critical than ever to establish security guidelines for new and evolving military systems, especially for information technology based systems. Software vulnerabilities, malicious code, and software that does not perform as intended pose an increased risk to the loss of operational capability and information superiority.

Software, in order to be useful, must be dependable (executes predictably and correctly under all conditions, including hostile conditions), trustworthy (contains few vulnerabilites or weaknesses that allow intentional loss of dependability or malicious behaviour of the software), and survivable (resilient to attack and able to recover quickly with minimal damages or loss of data from attacks it cannot resist). At a minimum, good secure software provides the following:

- Identification, **Authentication**, and **Authorization** to ensure proper control of access to the software and the data it handles
- **Confidentiality** to prevent unintended disclosure of information
- **Integrity** to ensure correctness and reliability of the software along with **information assurance** to provide assertions that the software, and the data handled by it, are used correctly
- Availabiliy to ensure the software is able to be used when required
- Management capabilities to manage and audit the use of the software

Software security requires active consideration throught the lifecycle to include the requirements, development, deployment, operation, and substainment phases.

The detailed perspectives listed below provide guidance for the development of secure software organized around two security aspects that apply to the development of any software system. The first aspect is the technologies and standards used to enable security, and the second is the policies and processes which promote security.

In addition to these detailed perspective, two references are provided as additional resources. The *Information Assurance Technology Analysis Center (IATIC) Software Security Assurance: A State-of-the-Art Report (SOAR)* [R1338] provides techniques (to include process models, life cycle models, and best practices) useful for the production secure software. The report *Software Assurance in Acquisition: Mitigating Risks to the Enterprise* [R1340] provides processes and guidance useful for both software practitioners and acquisition personel to ensure the development of software that is secure.

Detailed Perspectives

- Technologies and Standards for Implementing Software Security [P1391]
- Policies and Processes for Implementing Software Security [P1392]

P1391: Technologies and Standards for Implementing Software Security

The following perspectives provide guidance and best practices regarding the role of technologies and standards for implementing software security in the following areas:

- Using Public Key Infrastructure (PKI) related technologies to enable identification, authentication, and authorization
- Using XML Digital Signatures to provide non-repudiation
- Using encryption technologies and guidance to provide confidentiality
- Providing secure services
- Protecting data storage
- Using programming languages securely

Detailed Perspectives

- Public Key Infrastructure (PKI) and PK Enable Applications [P1061]
- Key Management [P1041]
- Certificate Processing [P1009]
- Smart Card Login [P1315]
- XML Digital Signatures [P1387]
- Encryption Services [P1020]
- SOAP Security [P1085]
- Security Assertion Markup Language (SAML) [P1189]
- RDBMS Security [P1064]
- LDAP Security [P1042]
- JNDI Security [P1039]
- Application Resource Security [P1005]
- Java Security [P1038]

P1061: Public Key Infrastructure (PKI) and PK Enable Applications

More and more secure **client/server** applications are appearing on the market. Applications today are relying heavily on **Digital Signature** technology to certify messages received were indeed sent by the sender. Both of these technologies use **Public Key encryption**, which is currently the only feasible way of implementing security over an insecure network such as the **NIPRNet**. Public Key encryption ensures that any form of communication that many contain sensitive information (i.e., passwords, credit card numbers) is protected while in transit and provides assurance to the receiver that the message was really sent by the sender. In the case of Web-based technologies, this is accomplished with a server that implements **encryption** at the communications level. The de facto standards for communication based encryption are the **Secure Sockets Layer** (**SSL**) and **Transport Layer Security** (**TLS**) protocols. The infrastructure used to support communication-based encryption is **PKI** which is composed of a number of cryptographic technologies but provides for two key services, data integrity and confidentiality. **Public Key** systems involve a **Certificate Authority** (**CA**) responsible for issuing a pair of digital **certificates**: one public and one private. The public key, as its name suggests, may be freely disseminated. This key does not need to be kept confidential. The **Private Key**, on the other hand, must be kept secret. The owner of the key pair must guard the private key closely, as sender authenticity and non-repudiation are based on the signer having sole access to the private key. There are several important characteristics of these key pairs. First, while they are mathematically related to each other, it is impossible to calculate one key from the other. Therefore, the private key cannot be compromised through knowledge of the associated public key. Second, each key in the key pair performs the inverse function of the other. What one key does, only the other can undo.

The CA is a trusted third party that issues digital certificates to its subscribers, binding their identities to the key pairs they use to sign electronic communications digitally. Digital certificates contain the name of the subscriber, the subscriber's public key, the digital signature of the issuing CA, the issuing CA's public key, and other pertinent information about the subscriber and the subscriber's organization. The CA can revoke certificates upon private key compromise, separation from an organization, etc. These certificates are stored in an on-line, publicly accessible repository. The repository, referred to as **Certificate Revocation List** (**CRL**), also maintains an up-to-date listing of all revoked but not yet expired certificates.

For the DoD PKI, users interface with the ***Real Time Automated Personnel Identification System (RAPIDS)*** workstation via the ***Issuance Portal*** for digital certificates residing on the **Common Access Card** (**CAC**).

To guarantee that data stays confidential and secure from attackers listening on the network in promiscuous mode (i.e., network sniffers) and to provide better performance, **Symmetric Encryption** (secret key) is used to encrypt and decrypt the data. **Asymmetric Encryption** (public key-private key) is not used for all encryption because it is too expensive for high volume data. For SSL and TLS, Asymmetric Encryption is used initially to pass the **secret key** (often called the **session key**). Once the secret key has been established on both sides, all subsequent data communications can be performed using Symmetric Encryption.

There are at least two options when an application needs to support PKI/SSL: use a DoD-approved **module** or develop the application abiding by the **DoD Class 3 Public Key Infrastructure Interface Specification**. The guidance linked to this perspective applies to **Public Key Enabled** applications wanting to operate within the DoD PKI.

Guidance

- G1308: Configure **Public Key Enabled** applications to use a **Federal Information Processing Standard** (**FIPS**) 140-2 certified cryptographic module.
- G1309: Make applications handling high value unclassified information in Minimally Protected environments **Public Key Enabled** to interoperate with **DoD High Assurance** .
- G1310: Protect application cryptographic objects and functions from tampering.
- G1311: Use **Hypertext Transfer Protocol over Secure Socket Layer** (**HTTPS**) when applications communicate with DoD **Public Key Infrastructure** (**PKI**) components.
- G1312: Make applications capable of being configured for use with DoD **PKI**.
- G1313: Provide documentation for application configuration for use with DoD **PKI**.

P1041: Key Management

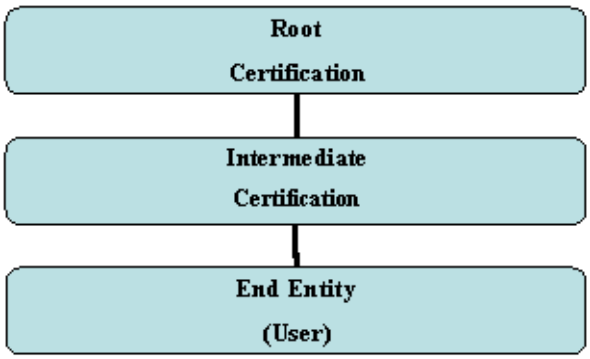
The key enabler in the **PKE** applications is **Asymmetric Encryption**, the use of **public** and **private keys**. It is used in exchanging **session keys**, and it is used to verify **Certificates** therefore, it is critical for applications to manage and protect the keys used in **PKI**. This includes the associated technologies used to store the keys and Certificates. The following list of guidance addresses key management issues.

Guidance

- G1314: Provide applications the ability to import and export keys (software certificates only).
- G1316: Ensure that applications protect **private keys**.
- G1317: Ensure applications store **Certificates** for subscribers (the owner of the **Public Key** contained in the Certificate) when used in the context of signed and/or encrypted email.
- G1318: Develop applications such that they provide the capability to manage and store **trust points** (**Certificate Authority** Public Key **Certificates**).
- G1319: Ensure applications can recover data encrypted with legacy keys provided by the DoD **PKI** Key Recovery Manager (**KRM**).

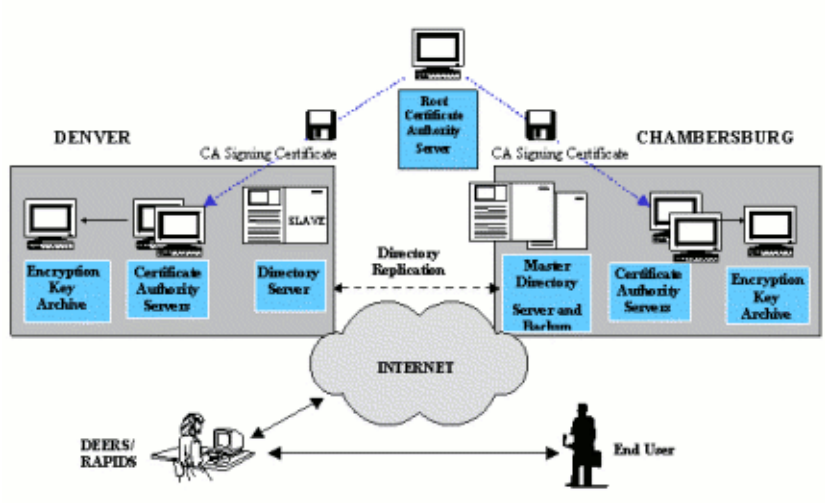
P1009: Certificate Processing

The **DoD** implementation of the **Public Key Infrastructure** (**PKI**) is the framework and services that provide for the generation, distribution, control, tracking and destruction of **Public Key Certificates**. The purpose of a PKI is to manage keys and **Certificates** in a way whereby the DoD can maintain a trustworthy networking environment. Digital Certificates are issued by a DoD **Certificate Authority**. It is an electronic document that contains a user's **identity**, a pubic key, a validity period, and the issuing authority. It is digitally signed and the Certificate is chained hierarchically in a path that can be traced to the Root Certificate.



I1091

Certificates can be sent via email or more commonly retrieved from repositories (**Directory Server**). Applications must validate the Certificate by checking status of the Certificate. There are two forms of status checking, the legacy Certificate Revocation List (**CRL**) or **Online Certificate Status Protocol** (**OCSP**). The status check determines whether a Certificate is revoked. A Certificate can be revoked if the information in the Certificate may have changed (relocation, new email) or the Certificate has been compromised. The Certificate validation is a critical part of the PKI process; it is the application's responsibility to perform the status checks. The following guidance sets the guidelines for the Certificate processing.



I1093

Guidance

- G1327: Enable an application to obtain new **Certificates** for subscribers.
- G1328: Enable an application to retrieve **Certificates** for use, including relying party operations.
- G1330: Ensure applications are capable of checking the status of **Certificates** using a **Certificate Revocation List** (**CRL**) if not able to use the **Online Certificate Status Protocol** (**OCSP**).

- G1331: Ensure applications are able to check the status of a Certificate using the **Online Certificate Status Protocol** (**OCSP**).
- G1333: Only use a **Certificate** during the Certificate's validity range, as bounded by the Certificate's "Validity - Not Before" and "Validity - Not After" date fields.
- G1335: Make applications capable of being configured to operate only with PKI Certificate Authorities specifically approved by the application's owner/managing entity.
- G1338: Ensure that **Public Key** Enabled applications support multiple organizational units.

P1315: Smart Card Logon

Smart Card Logon (SCL), also called Cryptographic Logon (CLO), capability enables users to log onto their unclassified network using their **Common Access Card** (**CAC**) and associated Personal Identification Number (PIN) instead of a username and password.

This capability addresses the Department of Defense (DoD) mandate in DoD Instruction 8520.2 [R1206] to Public Key (PK) enable all unclassified networks for certificate-based authentication to DoD information systems. SCL provides the increased security of two-factor authentication by allowing users to access their network with something they have (their CAC with DoD issued certificates) and something they know (their PIN).

> ***Note:*** *Joint Task Force-Global Network Operations (JTF-GNO) Communications Tasking Orders (CTOs; for example, CTO 06-02 and CTO 07-015) provide specific implementation directions for DoD, to include non-Windows-based operating systems (see https://www.jtfgno.mil/index.htm; DoD PKI required). Additional Mobile Code policy information is available from the* ***Information Assurance Support Environment*** *Web site, https://iase.disa.mil/mcp/index.html; DoD PKI required.*

Before enabling SCL, each unclassified network must also meet the following requirements:

- Implement **Active Directory** in the root domain
- Equip user workstations with a DoD-approved Windows operating systems, smart card readers, drivers, and the appropriate version of middleware
- Populate Active Directory accounts with each user's **Electronic Data Interchange Personal Identifier** (**EDI-PI**) numbers associated with the **CAC** certificates

Once users start using SCL to access their unclassified networks, they no longer need to remember their ever-changing and complex network passwords. SCL is a more secure method of network logon because the PIN is not stored on or transmitted over the network.

The following process illustrates how to use the PKI certificate for network logon:

- The user inserts the user's CAC into the smart card reader attached to the workstation, and, when prompted, enters the user's CAC PIN instead of a username and password
- A secure process retrieves the PKI certificate from the CAC and verifies it is valid and from a trusted issuer
- The user's workstation verifies the network domain controller's certificate is valid and from a trusted issuer
- If the user's PKI certificate and the domain controller certificate are valid, the user is automatically logged onto the network

> ***Note:*** *There are certain user groups (e.g., system administrators) that are unable to use PKI Certificates on a CAC as the primary token for smart card logon. A DoD CIO memo of 14 August 2006, Approval of the Alternate Logon Token (available via Defense Knowledge Online, https://www.us.army.mil/ [user account and DoD PKI Certificate required] DoD PKE Knowledge Base Library Smart Card and Alternate Token folders) permits the use of an Alternate Logon process.*

The Defense Manpower Data Center (DMDC) Common Access Card site (http://www.dmdc.osd.mil/smartcard) contains additional information, reports and developer support concerning the DoD CAC implementation.

Guidance

- G1862: Configure **Active Directory** for **Smart Card** Logon.
- G1869: Configure Domain Controllers for **Smart Card** Logon.

P1387: XML Digital Signatures

XML signatures are a form of **digital signatures** applied to digital content including XML; XML signatures are represented as XML, but the signed data may be any collection of digital content. XML signatures are usually used to sign XML documents or portions thereof. XML signatures as defined in NESI, particularly in this perspective, are specified by the W3C recommendation *XML Signature Syntax and Processing*.

XML signatures often serve as electronic versions of signatures. XML signatures provide a means to implement non-repudiation and detect changes to signed content.

Signing XML content is more complicated than signing other digital content, since XML has more than one syntactically correct way to express data. Because digital signatures are based on a hash of the signed content, a singe byte difference in the signed content can cause a verification of the digital signature to fail. The following examples show ways to represent different syntactically correct XML documents that may be semantically equivalent in a given context.

- White space is often insignificant within XML documents (<Node > is syntactically identical to <Node>)
- Order of XML attributes may vary
- Nodes within an XML document may have different XPath representations (for example using a relative path versus an absolute path)
- Namespace prefixes may have different name but represent to same namespace
- Namespaces declarations may occur in any order
- XML Element attributes may vary in order
- Child elements may inherent namespaces from parent elements which creates portability issues for signed nodes that are moved from one XML document to another
- Line break characters may vary between operating systems
- Order of XML nodes can vary or be unspecified
- XML comments may vary between XML documents

Because XML allows these different representations within XML documents, it is necessary to conduct a **canonicalization** of the XML document before signing a XML document and before verifying a signature of an XML document. Unfortunately existing canonicalization specifications are insufficient in some case and impact the interoperability and use of XML digital signatures. In some cases, it is necessary for developers to conduct their own canonicalization of XML as a precondition before signing the XML and again before verifying the signature of the signed XML to ensure consistency between the signed and verified documents and to account for inconsistencies for which the current canonicalization specification do not account.

In addition to issues relating to canonicalization and signature creation and verification, there is a potential to abuse digital signatures to conduct denial of service, cross-site scripting, or replay attacks through the use of carefully crafted XSLT and XPath expressions. To work around these issues, developers often employ a number of best practices to limit or reduce the impacts of such attacks. The W3C is drafting a collection of such best practices for the practical and secure use of XML digital signatures: http://www.w3.org/TR/xmldsig-bestpractices/. In addition to these best practices, NESI provides a number of guidance and best practice statements for the use of XML digital signatures.

The following links provide additional information for XML Digital Signatures and Canonicalization specifications.

- W3C Recommendation, *XML Signature Syntax and Processing (Second Edition)*, 10 June 2008, http://www.w3.org/TR/xmldsig-core/
- W3C Recommendation, *Canonical XML Version 1.1*, 2 May 2008, http://www.w3.org/TR/2008/REC-xml-c14n11-20080502/
- W3C Recommendation, *Exclusive XML Canonicalization Version 1.0*, 18 July 2002, http://www.w3.org/TR/2002/REC-xml-exc-c14n-20020718/

Guidance

- G1366: Digitally sign all **messages** where non-repudiation is required.
- G1367: Digitally sign **message** fragments that are required not to change during transport.
- G1371: Use the **Digital Signature Standard** for creating **Digital Signatures**.
- G1902: Use the Exclusive Canonicalization algorithm when digitally signing **XML** content that may be embedded in another XML document.

Best Practices

- BP1903: Include an xsd:dateTime field within long-lived **XML** digital signatures.

P1020: Encryption Services

Successful implementation of **Public Key** enabled applications is predicated on the correct selection and use of security algorithms. This section provides guidance on the use of **encryption**, **digital signature**, and authentication services in a consistent manner to interoperate with DoD **PKI**.

Guidance

- G1320: Use a minimum of 128 bits for **symmetric keys**.
- G1321: Enable applications to be capable of performing **Public Key** operations necessary to verify signatures on DoD **PKI** signed objects.
- G1322: Ensure that applications that interact with the DoD **PKI** using **SSL** (i.e., **HTTPS**) are capable of performing cryptologic operations using the **Triple Data Encryption Algorithm** (**TDEA**).
- G1323: Generate random **symmetric encryption** keys when using symmetric encryption.
- G1324: Protect **symmetric keys** for the life of their use.
- G1325: Encrypt **symmetric keys** when not in use.
- G1326: Ensure applications are capable of producing Secure Hash Algorithm (**SHA**) **digests** of **messages** to support verification of DoD **PKI** signed objects.
- G1797: Use a minimum of 1024 bits for **asymmetric keys**.

P1085: SOAP Security

Several security challenges arise from implementing a typical **service-oriented architecture** using **SOAP** including the following:

- **Authentication** (ensure that the sender of the **message** is genuine)
  - Preventing **identity** spoofing when accessing to a Web service.
  - Preventing tampering with the **WSDL** file of a Web service provider in order to spoof an endpoint.
- **Integrity** (ensure that an unauthorized third party cannot change a message during transmission without detection)
  - Preventing the interception of a message to or from a Web service provider to change its contents.
- **Confidentiality** (ensure that a message cannot be read by an unauthorized third party during transmission)
  - Preventing the interception of a message to or from a Web service provider and to obtain privileged information.

These security challenges are commonly addressed at the communication layer, the message layer, or both. The **Secure Sockets Layer** (**SSL**) and **Transport Layer Security** (**TLS**) protocols are commonly applied to the communication layer to provide confidentiality and authentication (both one-way and two-way authentication of service producers and consumers); see the Authorization and Access Control [P1339] perspective for further information.

Industry standards organizations such as the **World Wide Web Consortium** (**W3C**) and **Organization for the Advancement of Structured Information Standards** (**OASIS**) address these threats at the message level by specifying standards for providing authentication, protecting integrity and ensuring confidentiality. A common set of message layer specifications in the SOAP security space includes the following:

- Web Services Security (WS-Security) provides message layer mechanisms for implementing SOAP security. WS-Security supports message integrity through the use of XML Digital Signatures, support message confidentiality through the use of XML Encryption, and support authentication through the use of credentials such as X.509 certificates, **Security Assertion Markup Language** (**SAML**) tokens, and username/passwords.
- XML Digital Signatures provide a means to implement non-repudiation and detect changes to signed content. See the XML Digital Signatures [P1387] perspective for additional information.
- XML Encryption provides confidentiality by specifying a process for encrypting data (arbitrary data to include XML content). The result of the encryption processes is an XML element containing or referencing the encrypted data. XML Encryption can be selectively applied to data (for example to only parts of a XML document).
- SAML specifies ways to exchange security information (such as authentication, authorization, and attribute information related to assertions) across security domains. See the Security Assertion Markup Language [P1189] perspective for more information.
- **eXtensible Access Control Markup Language** (**XACML**) is a specification used in conjunction with SAML to represent and exchange access control policies across an enterprise.
- Web Services Policy (WS-Policy) describes a model and syntax for Web services to describe its requirements (required security policies, supported encryption algorithms, message delivery reliability requirements, etc.).
- WS-Trust specifies ways to issue, renew, obtain, and validate security tokens used to create trust relationships between participants in a secure message exchange.

Guidance

- G1356: Use the **SOAP** standard for all **Web services**.
- G1357: Do not rely solely on transport level security like **SSL** or **TLS**.
- G1359: Bind **SOAP Web service** security policy assertions to the service by expressing them in the associated **WSDL** file.
- G1362: Validate XML messages against a **schema**.
- G1363: Do not use clear text passwords.

- G1364: Hash all passwords using the combination of a timestamp, a **nonce** and the password for each **message** transmission.
- G1365: Specify an expiration value for all security tokens.
- G1366: Digitally sign all **messages** where non-repudiation is required.
- G1367: Digitally sign **message** fragments that are required not to change during transport.
- G1369: Digitally sign all requests made to a security token service.
- G1371: Use the **Digital Signature Standard** for creating **Digital Signatures**.
- G1372: Use an X.509 **Certificate** to pass a **Public Key**.
- G1373: **Encrypt messages** that cross an **IA** boundary.
- G1374: Individually **encrypt** sensitive **message** fragments intended for different intermediaries.
- G1376: Do not **encrypt** message fragments that are required for correct **SOAP** processing.

Best Practices

- BP1360: Use the **XML** Infoset standard to serialize messages.
- BP1375: Use **asymmetric encryption** for sensitive **SOAP**-based **Web services**.

P1189: Security Assertion Markup Language (SAML)

The Security Assertion Markup Language (SAML) is a vendor-neutral protocol specification for software applications and services to exchange security information in a distributed network environment. The SAML specification, maintained by the **OASIS** Security Services Technical Committee, defines schemas for how security assertions are structured and embedded within transport protocols.
SAML defines three types of assertions for an individual or machine:

Authentication	used for proving identity
Authorization	used for controlling access
Attributes	used to provide additional details to constrain the request

Email address, employee number, and rank are examples of attribute assertions.
SAML does not define any implementation of the services that authenticate or authorize users. Commercial vendors provide implementations in the form of authentication servers to authenticate and authorize users. Authentication servers respond to SAML requests and return SAML assertions that ensure the subject is logged in and authorized to access the resource.

Guidance

- G1379: Use **SAML** version 2.0 for representing security assertions.
- G1380: Use the **XACML** 2.0 standard for **SAML**-based rule engines.

P1064: RDBMS Security

Relational Database Management Systems remain on top amidst emerging technologies such as **XML** and **Object-Oriented Database Management Systems**. The continued dominance of **relational databases** is unlikely to change in the near future. First, there is still a large amount of legacy data and legacy applications that rely on **RDBMS**. Second, RDBMS are continuing to evolve to integrate XML as a function of the database. RDBMS is a reliable and proven technology that will be here for the long run. This perspective provides guidance on how best to secure the database.

Guidance

- G1346: Audit database access.
- G1347: Secure remote connections to a database.
- G1348: Log database **transactions**.
- G1349: Validate all input that will be part of any dynamically generated **SQL**.
- G1350: Implement a strong password policy for **RDBMS**.
- G1351: Enhance database security by using multiple user accounts with constraints.
- G1352: Use database clustering and redundant array of independent disks (RAID) for high availability of data.

Best Practices

- BP1355: Do not design the database around the requirements of an application.
- BP1353: Use a data abstraction layer between the RDBMS and application for externally-visible applications to prevent the disclosure of sensitive data.

P1042: LDAP Security

The **Lightweight Directory Access Protocol** (**LDAP**) can be thought of as a datastore. It is an open Internet standard produced by the **Internet Engineering Task Force** (**IETF**). LDAP is, like X.500, both an information model and a protocol for querying and manipulating it. The LDAP overall data and namespace model is essentially that of X.500. The major difference is that the LDAP protocol itself is designed to run directly over the **TCP/IP** stack, and it lacks some of the more esoteric DAP protocol functions. LDAP can store text, photos, **URLs**, pointers to whatever, binary data, and Public Key **Certificates**.

Guidance

- G1377: Use **LDAP** 3.0 or later to perform all connections to LDAP repositories.
- G1378: Encrypt communication with **LDAP** repositories.

P1039: JNDI Security

The **Java Naming and Directory Interface** (**JNDI**) is an **API** for directory services in a **Java EE** environment. It allows **clients** to discover and look up data and objects using a name. JNDI is portable and independent of the actual implementation. Additionally, it specifies a **service provider** interface (SPI) that allows plugging **directory service** implementations into the framework. The JNDI service implementations are hidden from the user and may make use of a **server**, a flat file, or a database. The choice is up to the JNDI provider.

Guidance

- G1071: Use vendor-neutral interface connections to the enterprise (e.g., **LDAP**, **JNDI**, **JMS**, databases).
- G1079: Isolate tailorable data values into the **deployment descriptors** for **Java EE** applications.
- G1239: Use **design patterns** (e.g., **facade**, **proxy**, or **adapter**) or property files to isolate vendor-specifics of vendor-dependent connections to the enterprise.

Best Practices

- BP1116: If using **Java**-based messaging (e.g., **JMS**), register destinations in **Java Naming and Directory Interface** (**JNDI**) so **message clients** can use JNDI to look up these destinations.

Examples

```
// Step 1
// Create a hashtable that contains the parameters
// used to initialize JNDI.
Hashtable contextParams = new Hashtable();
// Step 2
// Specify the context factory to use. The context
// factory is provided by the
// implementation.
contextParams.put( Context.INITIAL_CONTEXT_FACTORY, "com.jnidprovider.ContextFactory");
// Step 3
// The next parameter is the URL specifying the location
// of the JNDI provider's data store
contextParams.put( Context.PROVIDER_URL, "http://jndiprovider-database");
// Step 4
// Create the JNDI provider's context.
Context navyCurrentContext= new InitialContext ( contextParams );
// Step 5
// Look up the desired bean using its full name.
Object reference= navyCurrentContext.lookup ( "mil.us.navy.NavyBean" );
// Step 6
// Cast the located bean to the desired type.
MyBean navyBean= (NavyBean) PortableRemoteObject.narrow ( reference );
```

P1005: Application Resource Security

Applications use and store a large amount of data that often do not go into databases. For instance, an application often uses configuration files for application configuration, preferences files for personalization information (custom user experience) and resource files for internationalization support. Apply appropriate protection to sensitive resources to prevent attackers from tampering. Application bundles, properties files, configuration files when tampered could cause the user to execute inappropriate commands, expose sensitive data due to invalid configuration or cause the application to be inoperable. Therefore, it is of utmost importance to take appropriate measures to protect these resources.

Guidance

- G1344: Encrypt sensitive data stored in configuration or resource files.

P1038: Java Security

Java is an **Object Oriented Language**; applications benefit from the encapsulation features which offers protection for application data. Java was also designed and built with security in mind. Some of the security features include restricting direct access to memory (protecting data access privileges), array bounds checking (buffer overflow), and ability to install a security manager to protect resources. Despite all the security features built into the Java language, it does not mean that Java **APIs** are immune to security problems. Take care in the design and implementation of APIs to prevent attacks. The following security guidance are targeted to Java-specific APIs.

Guidance

- G1341: Use a security manager support to restrict application access to privileged resources.
- G1342: Restrict direct access to class internal variables to functions or methods of the class itself.
- G1343: Declare classes final to stop inheritance and prevent methods from being overridden.

P1392: Policies and Processes for Implementing Software Security

Many software errors and exploits share similar root causes resulting from the failure to follow common high level best practices. The detailed perspectives listed below provide best practices to enable compliance with policies and processes for implementing software security.

The Secure Coding and Implementation Practices [P1316] perspective provides a high level overview of important areas for consideration during software development from a programming language independent viewpoint. It discusses software security activities and best practices for use throughout the development lifecycle.

Protecting Data at Rest has become increasingly critical given Information Technology trends toward utilizing highly mobile computing devices and removable storage media. The Data at Rest [P1360] perspective provides guidance for complying with the DoD memorandum *Encryption of Sensitive Unclassified Data at Rest on Mobile Computing Devices and Removable Storage Media* [R1330] which mandates encryption not only for **Personally Identifiable Information** (**PII**) information but for all non-publicly released unclassified information contained on mobile computing devices and removable storage media.

The Mobile Code [P1314] perspective provides guidance to comply with DoD Instruction 8552.01, *Use of Mobile Code Technologies in DoD Information Systems* [R1292]. This Instruction identifies DoD-defined mobile code risk categories, describes their characteristics, and establishes restrictions for the acquisition (to include development) and use of mobile code technologies assigned to each risk category. This instruction applies to all DoD-owned or DoD-controlled information systems used to process, transmit, store, or display DoD information including mobile devices.

Detailed Perspectives

- Secure Coding and Implementation Practices [P1316]
- Data at Rest [P1360]
- Mobile Code [P1314]

P1316: Secure Coding and Implementation Practices

Many software errors and exploits share similar root causes resulting from the failure to follow common high level best practices. This perspective provides insight into a few of the major secure coding and implementation best practices from a programming language independent viewpoint.

This perspective does not provide all required guidance and best practices for secure software development. However, it does strive to provide a high level overview of important areas for consideration during software development. Finally, this perspective serves as a resource for additional information and tools for building secure software.

For best effectiveness, software security activities should occur throughout the development lifecycle. For example, security requirements (such as required roles, privacy requirements, accreditation requirements, etc.) are captured during the requirement phase of software system development. During the design phase, high level concepts such as defense in depth and principal of least privilege are applied. During actual development, programmers follow predefined development practices to include applying a coding standard. Finally, unit testing, regression testing, and peer reviews test the developed software for security vulnerabilities and policies.

Detailed Perspectives

The Secure Coding Practices perspective includes the following topic areas:

- Apply Principal of Least Privilege [P1317]
- Practice Defense in Depth [P1318]
- Apply Secure Coding Standards [P1319]
- Apply Quality Assurance to Software Development [P1320]
- Validate Input [P1321]
- Heed Compiler Warnings [P1322]
- Handle Exceptions [P1323]

P1317: Apply Principle of Least Privilege

To minimize risk and side effects due to possible security vulnerabilities, each process, function, or method within a software system should execute with the minimal set of privileges necessary to complete the action. To enable execution of code with the minimal set of privileges required, separate code requiring access to different resources or higher privileges. Whenever it is necessary to have an elevated permission level to complete an action, the elevated permission should be held for a minimum time. This approach reduces the chance that a security exploit can execute arbitrary code and minimizes the impact when an exploit occurs.

Best Practices

- BP1881: Separate code based on required privilege.
- BP1889: Minimize execution at elevated privilege levels to the shortest time required.

P1318: Practice Defense in Depth

A good practice to manage risk is to have multiple layers of defensive strategies. This reduces risk, since an exploit in one layer of defense may be stopped by another layer of defense and therefore eliminate or limit the consequences of the exploit.

As an example, a software system may use **Secure Sockets Layer** (**SSL**), **Public Key Infrastructure** (**PKI**), WS-Security along with **SOAP**, and provide security in integrity using database stored procedures, triggers and views.

Guidance

- G1301: Practice layered security.

Best Practices

- BP1922: Design systems to have security as a core capability.

P1319: Apply Secure Coding Standards

Develop to a documented coding standard for each target development language and platform to minimize the likelihood of security vulnerabilities caused by programmer error. This coding standard should include secure coding practices but may also include standards and policies that improve readability or maintainability.

Guidance

- G1215: Provide a coding standards document.

P1320: Apply Quality Assurance to Software Development

Quality assurance techniques are a useful tool in identifying and eliminating security vulnerabilities. Source code audits and peer reviews should be a regular activity during software development and maintenance along with normal testing activities.
To the extent possible, utilize automated tools to assist in verifying that code meets standards as defined in the applicable coding standard document. This will result a more repeatable process and shorten the time required for a peer reviews.

Guidance

- G1304: Unit test all code.

P1321: Validate Input

Proper input validation can eliminate many software vulnerabilities. Do not limit validation to the presentation tier; rather, all implementations of external facing modules should validate inputs prior to use. This can help prevent attacks including SQL Injection, Cross-Site Scripting, Buffer Overflows, and Denial of Service.
Validation may include checking lengths of input parameters to prevent buffer overflows. It may also include checking input against a list of allowed or disallowed characters to prevent execution of arbitrary code.

Guidance

- G1302: Validate all inputs.
- G1362: Validate XML messages against a **schema**.
- G1349: Validate all input that will be part of any dynamically generated **SQL**.
- G1032: Validate all input fields.
- G1147: Use **domain analysis** to define the constraints on input data validation.
- G1339: Practice defensive programming by checking all method arguments.

P1322: Heed Compiler Warnings

Many run time errors are detectable during the compilation process. Compiler warnings are often useful in detecting possible violations of syntax rules and mistakes introduced by developers which may lead to run time errors. For example, a compiler may warn about use of the assignment operator "**=**" instead of the equality operator "**==**" inside an **if** statement or warn about unchecked buffer assignment which could lead to a buffer overflow resulting in the execution of arbitrary code.
A good security practice to prevent many of these errors is to detect them at compile time by compiling code using the highest warning level available for the compiler. Compilers often have a warning option which enables additional warnings, for instance the GCC **-Wall** flag and the Java **-Xlint** option. In many cases, these options only enable the most common warnings and additional flags are required. Detailed understanding of the specific warning capabilities of a given compiler are necessary to ensure that all of the desired warnings truly are enabled.

Upon receiving an error from the compilation process, developers should modify the code to remove the deficiency or explicitly document the code stating the reason the code is valid but still produces a warning. Some programming languages and compilers contain syntax for documenting such exception to compiler warnings and suppressing the warning from the compiler output.

> ***Note:*** *Compiler warnings may vary depending on the compiler used and the target platform.*

Best Practices

- BP1890: Compile code using the highest compiler warning level available.
- BP1891: Develop code such that it compiles without compiler warnings.
- BP1892: Explicitly document exceptions for valid code that produces compiler warnings.

P1323: Handle Exceptions

Exception objects can convey sensitive information through their message or exception type. Translate information from exceptions to display meaningful information to users without displaying sensitive information from the exception. For example, do not expose the file layout of a system to a user through an exception thrown during file access. When necessary, catch and sanitize internal exceptions before re-propagating them to other parts of the system or displaying the exception to the user.

Guidance

- G1094: Catch all exceptions for application code exposed as a **Web service**.
- G1340: Log all exceptional conditions.

Best Practices

- BP1893: Return meaningful, but non-sensitive, information from exception handlers.

P1360: Data at Rest

Protecting **Data at Rest** (**DAR**) has become increasingly critical given Information Technology trends toward utilizing highly mobile computing devices and removable storage media. **Personally Identifiable Information** (**PII**) or sensitive government information stored on devices such as laptops, thumb drives and personal digital assistants (PDAs) is often unaccounted for and unprotected. This can pose a problem if the devices containing PII are compromised, lost, or stolen. This has generated negative media attention and potentially exposed sensitive information.

DAR technologies allow protection of data stored on mobile computing devices in the event of theft or other loss by way of encryption and password protection, thus enhancing **information assurance** (**IA**) posture. **DoD**, concerned not only with the loss of PII but with all unclassified data contained on mobile devices, issued a memorandum on 3 July 2007 entitled *Encryption of Sensitive Unclassified Data at Rest on Mobile Computing Devices and Removable Storage Media*.[R1330] This memo mandates encryption not only for PII records, but for all non-publicly released unclassified information contained on mobile computing devices and removable storage media. The cryptography used in the DAR technologies must be **National Institute of Standards and Technology** (**NIST**) Federal Information Processing Standard (**FIPS**) 140-2 compliant.

The DoD memo also mandates that all new computer assets procured to support the DoD enterprise include a **Trusted Platform Module** (**TPM**) version 1.2 or higher where such technology is available. TPM is a microcontroller that stores keys, passwords and digital certificates. It typically is affixed to the motherboard of computers. The nature of this hardware chip ensures that the information stored becomes more secure from external software attack and physical theft.

A U.S. General Services Agency (GSA) announcement on 14 June 2007 [R1334] notified **Chief Information Officers** (**CIOs**) that SmartBUY awarded Government-wide contractual agreements in May 2007 for DAR encryption commercial solutions to protect sensitive data. The GSA announcement identified contract awardees and provided a list of DAR encryption products available through the DoD SmartBUY Enterprise Software Initiative (ESI).

Guidance

- G1895: Encrypt all Unclassified DoD **Data at Rest** (**DAR**) not releasable to the public stored on mobile computing devices.
- G1896: Use **Data at Rest** (**DAR**) products that are **Federal Information Processing Standard** (**FIPS**) 140-2 compliant.
- G1897: Purchase **Data at Rest** (**DAR**) encryption products that are included in the Enterprise Software Initiative (ESI).
- G1381: Encrypt sensitive persistent data.

Best Practices

- BP1898: Purchase computers which contain a **Trusted Platform Module** (**TPM**).

P1314: Mobile Code

Mobile code is software obtained from remote systems, transferred across a network, and then downloaded and executed on a local system without explicit installation or execution by the recipient.

Conventional executable code refers to typical program code or software that is not embedded in data or text and that the user knowingly executes. Conventional executable code includes both compiled and interpreted code; examples include compiled C or Ada programs, scripts written in JavaScript or VBScript, Java applications, and binary .exe files.

Mobile code and ***active content*** are not interchangeable terms; incorrect usage can result in confusion. Mobile code is a broad term encompassing code obtained from a remote system that downloads across a network and executes on a local machine without the user's explicit initiation or knowledge. Active content is the term used to describe executable code embedded within (or bound to) text or data that executes automatically without explicit user initiation. Examples of active content include Microsoft Visual Basic for Applications (VBA) macros embedded in Microsoft Word and Excel files, PostScript commands embedded in PostScript documents, and scripts embedded in Macromedia Director and Shockwave movies.

As depicted in the figure below, mobile code is comprised of that active content or conventional executable code which has become "mobile." When active content and/or conventional executable code resides statically on the workstation or host on which it executes, it is not mobile code. However, when such code originates from an external system, traverses a network, downloads onto a workstation or host, and executes without explicit user initiation, it becomes mobile code.

I1218: Mobile Code

Mobile code brings many benefits to a computer system, such as reduction of communication, ability to perform asynchronous tasks, dynamic software deployment, and temporary and scalable applications. But despite all the benefits there are many threats that mobile agents bring to a computer system, such as denial of service, destruction, unauthorized access, breach of privacy, and theft of resources, among others. These threats are related to protection of the host systems and mobile code systems themselves.

The Department of Defense issued DoD Instruction 8552.01, *Use of Mobile Code Technologies in DoD Information Systems* [R1292], in October 2006 to establish and implement DoD mobile code policy. This Instruction identifies DoD-defined mobile code risk categories, describes their characteristics, and establishes restrictions for the acquisition (to include development) and use of mobile code technologies assigned to each risk category. It also establishes restrictions on the use of mobile code in email and emerging mobile code technologies and directs monitoring to detect the presence of prohibited mobile code. Any prohibited mobile code discovered must be removed.

This instruction applies to all DoD-owned or DoD-controlled information systems used to process, transmit, store, or display DoD information. This includes mobile devices (e.g., cellular phones, handheld devices) capable of executing mobile code. Mobile code that originates from and travels exclusively within a single enclave boundary is exempt from the requirements of DoD Instruction 8552.01. However, if an enclave consists of geographically dispersed computing environments that are connected by the **Non-Classified Internet Protocol Router Network** (**NIPRNet**), **Secret Internet Protocol Router Network** (**SIPRNet**), **Internet**, or a public network, the requirements of this instruction apply.

Category 1 Mobile Code

Category 1 mobile code technologies exhibit a broad functionality, allowing unmediated access to workstation, server, and remote system services and resources. Category 1 mobile code technologies have known security vulnerabilities with few or no countermeasures once they begin executing. Execution of Category 1 mobile code typically requires an all-or none decision: either execute with full access to all system resources or do not execute at all.

The following mobile code technologies are assigned to ***Category 1A*** (allowed):

- ActiveX controls
- Shockwave movies (including Xtras)

The following mobile code technologies are assigned to ***Category 1X*** (prohibited):

- Mobile code scripts that execute in Windows Scripting Host (WSH) (e.g., JavaScript and VBScript downloaded via a Uniform Resource Locator (URL) file reference or email attachment)
- HTML Applications (e.g., **`.HTA`** files) that download as mobile code
- Scrap objects
- Microsoft Disk Operating System (MS-DOS) batch scripts
- Unix shell scripts
- Binary executables (e.g., **`.exe`** files) that download as mobile code

The use of unsigned Category 1 mobile code in DoD information systems is prohibited.

Category 2 Mobile Code

Category 2 mobile code technologies have full functionality, allowing mediated or controlled access to workstation, server, and remote system services and resources. Category 2 mobile code technologies may have known security vulnerabilities but also have known fine-grained, periodic, or continuous countermeasures or safeguards.

The following mobile code technologies are currently assigned to ***Category 2***:

- Java applets
- Visual Basic for Applications (i.e., Visual Basic for Applications [VBA] macros)
- PostScript
- Mobile code executing in the Microsoft .NET Common Language Runtime
- PerfectScript
- LotusScript

Category 2 mobile code that does not execute in a constrained execution environment may be used in DoD information systems if the mobile code is obtained from a trusted source over an assured channel. Information regarding these assured channels is available from DoD Instruction 8552.01.

Category 3 Mobile Code

Category 3 mobile code technologies support limited functionality, with no capability for unmediated access to workstation, server, and remote system services and resources. Category 3 mobile code technologies may have a history of known vulnerabilities, but also support fine-grained, periodic, or continuous security safeguards.

The following mobile code technologies are currently assigned to ***Category 3***:

- JavaScript, including Jscript and ECMAScript variants, when executing in the browser
- VBScript, when executing in the browser
- Portable Document Format (PDF)
- Flash

Category 3 mobile code technologies may be freely used without restrictions in DoD information systems.

Emerging Mobile Code Technologies

Emerging mobile code technologies refer to all mobile code technologies, systems, platforms, or languages whose capabilities and threat level have not yet undergone a risk assessment and been assigned to one of the three risk categories described above.

Some examples of emerging technologies follow:

- Microsoft's .NET Framework, when used to execute mobile code
- The flat script files used by Java WebStart to control the execution of Java applications

Because of the uncertain risk, the use of emerging mobile code technologies in DoD information systems is prohibited.

Mobile Code in Email

Mobile code can be embedded in an email body or an email attachment and can be downloaded as part of the actual email. Alternately, mobile code residing on a remote server can be referenced from within an email body or attachment and can be automatically downloaded and executed. Some types of mobile code execute automatically as soon as the user clicks on the message subject or previews the message; others execute when the user opens an attachment containing mobile code. Email viruses, worms, and Trojan horses typically utilize mobile code technologies; they are forms of malicious mobile code sent to users via email.

Due to the significant risk of malicious mobile code downloading into user workstations via email, and the ease of rapidly spreading malicious mobile code via email, the following restrictions apply to all types of mobile code in email independent of risk category:

- To the extent possible, the automatic execution of all categories of mobile code in email bodies and attachments is disabled, compliant with DoD mobile code policy implementation guidance.
- To the extent possible, mobile code-enabled software is configured to prompt the user prior to opening email attachments that may contain mobile code.

Code-Signing Certificate Requirements

DoD code-signing certificates (i.e., their associated private keys) are used to sign Category 1A mobile code that will reside on DoD-owned or DoD-controlled servers prior to its installation on the servers. When code signing is used to meet the requirements for Category 2 mobile code that will reside on DoD-owned or DoD-controlled servers, the mobile code is signed with DoD code-signing certificates prior to its installation on the servers. DoD code-signing certificates are designated as trusted by default by all Components. DoD-owned and DoD-controlled servers are trusted sources by default.

Guidance

- G1883: Use a DoD PKI code signing certificate to sign mobile code residing on DoD-owned or DoD-controlled servers.
- G1884: Configure browsers to use Category 1A allowed mobile code per DoD Instruction 8552.01. [R1292]
- G1885: Configure browsers to disable Category 1X prohibited mobile code per DoD Instruction 8552.01. [R1292]
- G1886: Disable automatic execution of mobile code in email clients.
- G1887: Monitor configured mobile code-enabled software to ensure it is in compliance with DoD Instruction 8552.01. [R1292]

Best Practices

- BP1888: Only enable plaintext viewing in email clients on DoD-owned and DoD-operated information systems.

P1371: Security Services

This service area relates to security services necessary to protect sensitve information in the information system. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- Software Security [P1065]
- Enterprise Security [P1332]
- Network Information Assurance [P1147]

P1332: Enterprise Security

Security is not a single idea, object, or task. The common phrase ***defense in depth*** is very apt in describing how to secure **information technology** (**IT**) environments. While the objective may be to impede adversaries completely, slowing them down is the more likely and practical outcome. Some examples include the following:

- Causing an adversary to expend more resources to accomplish the same task
- Generally creating more exposure to enable better detection and disruption of an adversary's activities

Multiple security boundaries provide protection depth. Some of these boundaries are physical, while others are information-based in nature (e.g., virtual technologies, social processes or extended-trust meta-data). A heterogeneous approach is necessary for everything in a Node that must be protected, in order not to expose a single point of failure. The "weakest link" adage is very applicable to net-centric operational security (OPSEC).

Enterprise Security includes the fundamental core or "capstone" concepts and guidance for Security that are necessary to understand the "Security Considerations" found in the other Node functional environment perspectives. For a further discussion of security concerns regarding accountability, logging and auditing see the Enterprise Management [P1330] perspective.

Detailed Perspectives

- Cryptography [P1333]
- Integrity [P1334]
- Identity Management [P1178]
- Authorization and Access Control [P1339]
- Confidentiality [P1340]
- Network Information Assurance [P1147]
- Trusted Guards [P1150]

Guidance

- G1301: Practice layered security.

P1333: Cryptography

Cryptography is a fundamental technique to support operations security (OPSEC) by enabling the following activities:

Ensuring Integrity (e.g., digital signatures): Digital signatures enable tamper detection and non-repudiation. A digital signature or digital signature scheme is a type of cryptography used to simulate the security properties of a handwritten signature on paper with all the benefits and more. Optionally, include a scanned copy of the written signature for completeness. They cannot be copied or as easily forged. Digital signature schemes normally provide two algorithms, one for signing which involves the user's secret or **private key** (the only key in symmetric schemes), and (in asymmetric schemes) one for verifying signatures which involves the user's **public key**. The output of the signature process is called the "digital signature."

Authenticating identity (e.g., keys) Authentication is the process of attempting to verify the digital identity of the sender of a communication such as a log in request. The sender being authenticated, often referred to as the principal, may be a person using a computer, a hardware device or a computer program. An anonymous credential, in contrast, only weakly establishes identity, together with a constrained right or status of the user or program.

Ensuring confidentiality: Encryption of the payload covers data, signatures, session keys, certificates for integrity, authentication, and authorization information.

Authorization (e.g., X.509 certificates, roles, and accounts): Perform authentication prior to authorization. Authenticated identities, even an anonymous identity, are necessary to perform successful authorization. Authorization grants the level of privileges (authorization) assigned to a particular authenticated identity. In most cases, anonymous or weak authenticated identities should have limited capabilities or level of authorization, such as read-only access to general access resources.

Cryptographic guidance requires a sensitivity/protection/performance trade off analysis. Factors to consider follow:

- shelf life of information (actionable, analysis)
- key and algorithm hardness
- key length and type (symmetry versus asymmetry)
- management procedure attack resistance and resilience
- cryptography overhead impact
- transport path bandwidth-delay product for handshaking and key distribution
- processor speed and memory for encryption/decryption algorithms
- storage space and access speed for encryption/decryption algorithms

Complexity of crypto management is defined by the following:

- key assignment and distribution
- authorization scope (delegation, transitive trust, revocation, etc.)
- accountability
- auditability

Guidance

- G1371: Use the **Digital Signature Standard** for creating **Digital Signatures**.
- G1376: Do not **encrypt** message fragments that are required for correct **SOAP** processing.
- G1325: Encrypt **symmetric keys** when not in use.
- G1317: Ensure applications store **Certificates** for subscribers (the owner of the **Public Key** contained in the Certificate) when used in the context of signed and/or encrypted email.
- G1344: Encrypt sensitive data stored in configuration or resource files.
- G1374: Individually **encrypt** sensitive **message** fragments intended for different intermediaries.

- G1378: Encrypt communication with **LDAP** repositories.
- G1381: Encrypt sensitive persistent data.

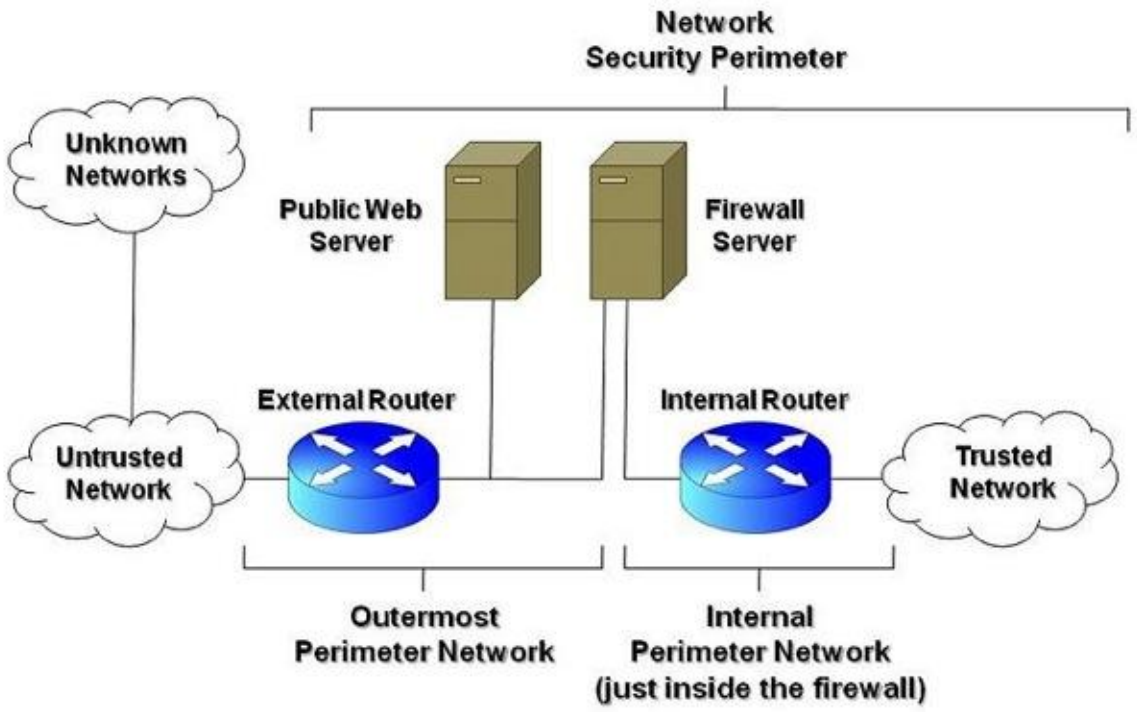
P1334: Integrity

Integrity of an **enterprise** consists of ensuring the overall integrity of its **systems** and the data they contain. External interfaces are the first line of defense, but defense-in-depth may require assurance controls on internal **Node** interfaces as well. A program's Capability Description Document (CDD) initially defines interfaces which the Node's architects formally specify. With proper safeguards and testing, interfaces can act as formal integrity boundaries.

Node and system architects ensure integrity by first specifying hardened boundaries and equipping them with sensors and security controls. Baseline vulnerability assessment information is also helpful. Vulnerability assessments should occur for every boundary interface that exposes and must protect data, applications and **services**. Evaluation of each interface will not only use net-centric metrics to indicate how well they make information available, but also by vulnerability metrics indicating how well they defend information within those boundaries. The following subsections and linked detailed perspectives cover the interface controls and security technologies that current **Information Assurance** (**IA**) guidance requires for each interface boundary. Not only do all boundary interfaces require interface controls, but the subsidiary boundary interfaces major architectural constructs provide require interface controls as well. Examples follow:

- computing infrastructure system boundaries and virtual machine boundaries
- transport network boundaries and subnetwork/overlay network/virtual network boundaries
- user environment boundaries and display or window boundaries
- management domain and sub-domain boundaries
- boundaries defined for the security technologies themselves, including subordinate **Certificate Authorities**
- data and service boundaries, including Web page frames, **applets** and **servlets**

The following diagram (I1239: *Example Two-Perimeter Network Security Design*) is an example of how to identify two such boundaries and their security control components. The diagram shows how to structure subsidiary boundaries in the Transport infrastructure in order to separate Nodes with different IA authorities and policies onto separate **Global Information Grid** (**GIG**) intra-networks, such as those found in joint operations. At the same time, by appropriate placement of transport routers and guards, the two services can interconnect and interoperate to coordinate their joint operations. This architectural structuring, because it is based on open standards, allows each service to select and standup its own implementation of the architecture, with its own security policies, without preventing the interoperable flow of authorized joint coordination information.



I1239: Example Two-Perimeter Network Security Design

Key security concepts are in the following subsections and the linked detailed perspectives. The security activities can serve as guides or templates for a Node's Interface Control Document (ICD), as required by the **Security Technical Implementation Guides** (**STIGs**) and the *DoD Information Assurance Certification and Accreditation Process (DIACAP)*.[R1291] The intent of these activities is to help Node architects and program managers determine the best ways to identify and mitigate weaknesses in Nodes while maintaining net-centric interoperability.

The subsections and the linked detailed perspectives also provide recommendations about how to select and apply the relevant standards and technologies to provide security capabilities. The intent is to mitigate the exposure of weak link systems in Nodes while maintaining interoperability. Certain security activities, techniques and technologies are common to among Node components.

- ***Integrity:*** Quality of an Information System (IS) reflecting the logical correctness and reliability of the operating system, the logical completeness of the hardware and software implementing the protection mechanisms, and the consistency of the data structures and occurrence of the stored data; formal security terminology often interprets integrity more narrowly to mean protection against unauthorized modification or destruction of information and does not require system behavior that meets all operational goals and expectations. [R1339]
- ***Defense-in-depth:*** establishes variable barriers across multiple layers and dimensions of networks. [R1339]
- ***Boundary:*** software, hardware, or physical barrier that limits access to a system or part of a system; [R1339] hardening techniques and technologies assure integrity and define security perimeters thanks to the embedding of security controls.
- ***Standard vulnerability specifications and scorecards based on them:*** examples include the Common Vulnerability Enumeration (CVE; see http://cve.mitre.org/ or http://en.wikipedia.org/wiki/Common_Vulnerabilities_and_Exposures), the Common [Software] Weakness Enumeration (CWE; see http://cwe.mitre.org/) and the Open Vulnerability Assessment Language (OVAL; see http://oval.mitre.org/); they help to

evaluate the hardness of boundary interfaces, the adequacy of the embedded security sensors or controls, and the effectiveness of the enterprise security engineering policies and support systems.

Security Integration Activities

The following security-based activities integrate security and IA throughout a Node using the above concepts. Each concept has a variety of techniques and technologies, use of which varies according to the functional category and Node operational requirements. The following sections are divided first into the functional categories, and then into the major activities. Specific techniques and technologies for that functional category's security activities are then listed as sub-sub-sections or lists.

Boundary Creation

Boundary creation includes selection of security control technologies to embed in boundary interfaces for baseline integrity protection. The simplest form often does not provide access control, just interoperability and accountability and in military settings is used primarily when physical boundaries and access control are sufficient assurance of Node integrity. When installing or embedding security controls, ensure the target Component is in a state of known integrity, e.g., by booting with known media such as Original Equipment Manufacturer (OEM) media or "gold" disks (referring to a master disk that has known safe status, documented chain of custody media, etc.). Also ensure that the components in question have valid anti-tamper signatures for their storage media, current malware signature files and scanner engines and very recently successfully completed holistic scans. See the Network Infrastructure Integrity [P1336] perspective and the DISA Information Assurance Support Environment (IASE) Security Technical Implementation Guides (STIGS) and Supporting Documents Web site for additional information.

Access Control Integration

Access control integration employs security controls (including, for example, identity management subsystems, virus scanners and guards) designed to detect and deny unauthorized access and permit authorized access in an IS.[R1339] This integration adds additional hardening as well as finer-grained control than the all or nothing access provided by simple boundary creation. However, interactions of these security controls with users and other principals, as well as with enterprise security systems, generate interoperability requirements and testing for the Node.

Quarantine Creation

Quarantine is the term which describes a special family of boundary-based damage control techniques and technologies that limit external compromises of systems to an in-Node isolation construct. These techniques often also provide a way to remedy identified deficiencies prior to re-enabling normal access to system resources. Also may provide additional boundary hardening to ensure the integrity of good Components missing necessary capabilities.

High Availability Integration

High availability integration is a configuration activity which assures with high probability that a system will be operational at any given time, and will recover quickly in the event of a failure. In general, a high-availability system has safeguards to prevent unscheduled outages from power failures, code defects, or hardware failures.

Management

In the security realm, management includes monitoring and configuring boundaries and their embedded security sensors/controls through use of enterprise security engineering support systems, operational policies and procedures.

Auditing

Most information systems have a logging facility and can log all "deny access" actions which would show intrusion attempts. Modern systems have an array of logging features that include the ability to set severity based on the data logged. An auditing schedule should be established to routinely inspect logs for signs of intrusion and probing.

Detailed Perspectives

- Computing Infrastructure Integrity [P1335]
- Network Infrastructure Integrity [P1336]
- User Environment Integrity [P1337]
- Data, Application and Service Integrity [P1338]

Guidance

- G1301: Practice layered security.
- G1300: Secure all **endpoints**.

P1335: Computing Infrastructure Integrity

Increasingly, security integration and enterprise security for the computing infrastructure is growing beyond securing basic hardware, firmware and software boundaries to include activities that must deal with boundaries based on virtual machines and **services** that cross system and **Node** boundaries. However, none of these more dynamic boundaries are secure unless the underlying basic components have the necessary integrity and other security capabilities.

The primary computing infrastructure boundary is the information **system** component. Subsidiary constructs include the firmware, the operating system (OS), the file system data storage, and application execution contexts such as the user account.

Operating System Hardening

Security of the operating system relies on creating some common boundaries. Creating these boundaries often requires numerous procedures such as configuring system and network interface components properly or removing or disabling unused, undefended and unnecessary files and services, while ensuring that all of the applicable security patches are in place. The DISA **Security Technical Implementation Guide** (**STIG**) repository contains authoritative checklists for operating system hardening. In addition there are Department of Defense Information Assurance Vulnerability Alert (IAVA) and Information Assurance Vulnerability Management (IAVM) notifications for compliance.

Data Storage Encryption

Data encryption can happen in many different ways. One method involves providing encryption as part of the storage. Many newer operating systems and applications have built in support for data encryption at the file, directory/folder, and volume/disk level. Each level has a potential need for boundary creation; this requires weighing the trade offs. For example, encryption at the folder or disk/volume level does not require that users or applications provide individual file encryption; therefore, auxiliary files receive automatic encryption support. However, finer-grained control will consequently require additional development, testing and training.

Remote data storage architectures typically perform encryption at the physical storage endpoint. Ensure that data remains encrypted when transmitted over the network to the physical storage endpoint to assure end to end confidentiality.

For further information see the Data at Rest [P1360] perspective.

DRM Signing at the OS and Hardware Level

Various operating systems and applications like Windows Vista, Windows Server 2008 and the Linux kernel 2.6.12 and later use Trusted Platform Module (TPM). TPM supports capabilities such as Windows BitLocker full-drive encryption technology as well as Digital Rights Management (DRM) and software licenses. A TPM microchip is embedded on the computer's (or other device's) motherboard and stores unique system identifiers along with the decryption keys. Certain systems may provide the TPM as part of the standard build.

Parity Checking

Beyond the standard use of parity checking performed with memory or communications there are also applications that make use of parity checking for the whole computer system such as Bit9. This is an example of one approach that can check a whole system for tampering to better protect against unanticipated (zero day) exploits, unauthorized software installations, etc. This process could be coded into proprietary software and or included into a program's Statement of Work (SOW), etc.

Virus Scanning

Viruses are a significant interface independent cross-boundary threat that requires constant monitoring. Some security control computing practices can help to mitigate the risk of virus infections and reduce the possibility of inadvertently triggering or spreading viruses and will help defend against malicious code attacks. Virus scanners are security controls and act as gatekeepers at boundaries. However, they do not require interoperability with other components or Nodes, except for enterprise security. Consequently, they do

not traditionally fall under the main capabilities associated with boundary gate-keeping, authorization or authentication.

Components should also enable baseline holistic scans of the whole system to prevent some of the stealthier viruses that can hide from any scan that is initialized while the system is already up and running.

Finally schedule anti-virus software to check in regularly with the master server that provides the signature and application updates.

For additional details see the Host Information Assurance [P1161] perspective.

Spyware and Malware Scanning

Spyware is a significant interface independent cross-boundary threat that requires repeated monitoring. In addition to enabling direct attacks, spyware is also a potential entry point for viruses. Enabling good security control placement can defend against malicious code attacks by limiting the risk of spyware infections, inadvertent triggering of, or spreading, spyware and related viruses.

Spyware security control programs share many best practices with related virus security control placement. Ensure that any spyware security control programs do not "step" on security control antivirus software and vice versa.

For additional details see the Host Information Assurance [P1161] perspective.

Computing Infrastructure Quarantine Support

Providing computing infrastructure quarantines is generally bundled with software security sensors and controls that detect unwanted or compromised software. With dynamic, service-oriented configurations, it is likewise important to have some type of spyware security sensor/control that can detect and remove or quarantine those unwanted "helper" components that repeatedly attempt to install themselves in a configuration. Quarantine is also a capability that is used by other security sensors/control components like malware scanners and analyzers.

High Availability

For more detailed guidance of highly available Computing Infrastructure, see DoD Instruction 8500.2, *Information Assurance (IA) Implementation*, [R1198] especially for Mission Assurance Category (MAC) I systems and networks. The following subsections summarize important concepts.

Data Backup and Recovery

Nodes should provide frameworks to support backup and recovery of data. Backup logs support auditing of activities.

Enable operations personnel to destroy backup media physically during disposal to prevent unauthorized reading of the media contents. Employ the "two person" rule to dispose of media; maintain meticulous tracking logs, available in hard copy as well as electronically, of all backup media.

Verify encryption of all data on removable media is with a level of encryption appropriate for the level of data protection required by policy.

Fault Tolerance

Critical components, ones on which other components are dependent such as enterprise services and infrastructure components, must not become weak links that significantly cripple the Node's operations. Their high availability ensures the continuity of operation. A precept of high availability architectures is that they are fault tolerant and/or redundant, starting with the hardware components. If a primary component fails, the secondary component takes over in a process that is seamless to the application running on the server. As such, fault-tolerant systems "operate through" a component failure without loss of data or application state.

In addition, fault tolerant/redundancy includes software-based failover clustering, in which a hardware or software failure on one server causes the workload to be shifted by the Computing Infrastructure to a second server.

Computing Infrastructure Configuration Rollback and Recovery

Nodes should provide frameworks to support backup and recovery of Node provisioning information to support configuration and change management activities. Nodes should make this framework available to Components to enable coordinated configuration and change management activities across all the Components in the Node.

Management

Management activities specific to the security realm have a heavy emphasis on managing cryptographic components of the computing infrastructure, especially those that provide key management.

Key Management

Key backup and recovery is especially important in data storage encryption to prevent loss of otherwise long-lifetime data. For example, if a disk is encrypted and then moved to another machine (because the original machine had a hardware failure), without good key backup and recovery, the data could be inaccessible. Designated key recovery agents should be kept to a minimum in order to expose fewer keys to cryptographic attack and provides a higher level of assurance that encrypted data will not be decrypted inappropriately. Refer to the National Institute of Standards and Technology Special Publication 800-57, *Recommendation for Key Management - Part 2: Best Practices for Key Management Organization (*NIST SP800-57-Part2) and the Key Management [P1041] perspective in NESI *Part 5* for additional information.

Auditing and Logging

Most information systems have a logging facility and can log all "deny access" actions which would show intrusion attempts. Modern information systems have an array of logging features that include the ability to set severity based on the data logged. An auditing schedule should be established to routinely inspect logs for signs of intrusion and probing.

Guidance

- G1622: Implement **commercial off-the-shelf** (**COTS**) software that protects against malicious code on each operating system in the Node in accordance with the Desktop Application **Security Technical Implementation Guide** (**STIG**).
- G1623: Implement personal **firewall** software on computers used for remote connectivity in accordance with the Desktop Applications, Network, and Enclave **Security Technical Implementation Guides** (**STIGs**).

Best Practices

- BP1707: Configure and locate elements of the Node Web infrastructure in accordance with the Web Server **Security Technical Implementation Guide** (**STIG**).
- BP1708: Configure and locate elements of the Node Web infrastructure in accordance with the Desktop Applications **Security Technical Implementation Guide** (**STIG**).
- BP1709: Configure and locate elements of the Node Web infrastructure in accordance with the Network **Security Technical Implementation Guide** (**STIG**).

P1336: Network Infrastructure Integrity

Network integrity is based on network boundaries and constructs that may not be as familiar to the average person as information system boundaries and constructs. Network boundaries and constructs are often the domain of network architects and operations rather than end users, and they are often not confined to a tangible system but distributed among multiple end systems, routers and switches. Network virtualization, for example, is a routine application of these principles. In many ways, however, network constraints are very much like computing infrastructure: there are hardware and software constructs whose boundaries must be hardened as a pre-requisite to securing more dynamic constructs such as **virtual private networks** (**VPNs**) and secure sessions.

Boundary Creation

Boundaries in Transport networks are a function of the physical, link and network layer technologies and are reflected in the address structures and the bindings. Aligning these Transport functional boundaries with **Information Assurance** (**IA**) boundaries and positioning the appropriate security controls is the subject of the following discussion.

The boundary between a host or router system and its local network is its network stack (or stacks, in routers); to be visible and reachable the boundary must have an **IP** address. Security controls at this boundary are primarily a function of hardening the system hardware and software, including the network stack.

Hardening a system is a combination of assuring initial integrity of the system and its default configuration through certification and accreditation processes such as the *DoD Information Assurance Certification and Accreditation Process (DIACAP)*.[R1291] Ongoing vulnerability management must follow, especially as system software changes and configurations are adapted to local requirements and policies.

The Network **Security Technical Implementation Guide** (**STIG**) on the DISA Information Assurance Support Environment (IASE) Web site provides guidance for the boundary between the **Node's** internal network and external networks. A summary and list of examples of what is in the Network STIG follows; see the Network Information Assurance [P1147] perspective for additional details.

Router Security Considerations

There are many things to consider when determining how to secure a router or other type of network device. They all involve using the router to support the appropriate placement of security sensors and controls to harden the various Transport boundaries. They also may require associated enterprise security components to manage the policies so deployed and enforced.

Patches and Updates

Subscribe to alert services provided by the manufacturers of any networking hardware so that they are up to date with both security issues and service patches. As vulnerabilities are found, and they inevitably will be found, good vendors make patches available quickly and announce these updates through e-mail or on their Web sites. Always test the updates before implementing them in a production environment.

Protocols

Denials of service attacks often take advantage of protocol-level vulnerabilities, for example, by flooding the network. To counter this type of attack, add Node security controls and policies.

- use ingress and egress filtering
- screen Internet Control Message Protocol (ICMP) traffic from the internal network
- block trace route
- control broadcast traffic
- block other unnecessary traffic

Ingress and Egress Filtering

Spoofed packets (packets with fake or hijacked addresses) are indicative of probes, attacks, and other activities by a knowledgeable attacker. Network boundary devices should verify both incoming and outgoing packet addresses. While this does not protect the Node from a denial of service attack, it does keep such attacks from originating from the Node's network and if other networks apply the same verification, the Node's network could be saved from a denial of service attack.

This type of filtering also enables the originator to be easily traced to its true source since the attacker would have to use a valid, and legitimately reachable, source address. For more information, see the Internet Engineering Task Force (IETF) *Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing* Request for Comment (RFC 2827).

ICMP Traffic

The Internet Control Message Protocol (ICMP) is a stateless protocol that uses the Internet Protocol (IP) and allows verification of host availability information from one host to another. It often is used for Enterprise Management performance testing and fault isolation. However, providing a security control that can block ICMP traffic at the outer perimeter router will protect the Node from cascading ping floods and other denial of service attacks.

Trace Route

Trace route is a means to collect network topology information. It detects devices en route to a destination system and is very useful in determining whether Node and mission data is traveling along optimal routes. Its implementation varies for each manufacturer; some use a ping with differing time to live (TTL) values while others use a **User Datagram Protocol** (**UDP**) datagram. Enabling policies that block ICMP messages can control the variable ping, while the UDP datagram may require an **access control list** (**ACL**) type policy to block it. By enabling the deployment of blocking policies of this type, security controls prevent an attacker from learning details about the Node's network.

Broadcast Traffic

Directed broadcast traffic can be used to discover and enumerate hosts on a network and as a vehicle for a denial of service attack. For example, by blocking specific source addresses, security controls prevent malicious echo requests from causing cascading ping floods.

Unnecessary Traffic

Incoming traffic from the **Internet** to the boundary router is from unknown, untrusted users who require access to the Node's **Web servers**. The users are accessing a specific list of IP addresses and port numbers and can be restricted to access no other port numbers or IP addresses. Using access control lists (security controls available on most routers) only traffic for the desired combination of addresses and ports can pass through the boundary router; an assumption is that any other addresses are potentially hostile. Port numbers in this example are not related to ports on a switch which are the physical sockets into which the Ethernet cables are plugged. Here, the reference is to the IP addressing system, where the IP address is extended with a **TCP** or UDP port number. For example a Web server is frequently on port **`80`**; the full address of the Web service on a server with an IP address of **`192.168.0.1`** would be **`192.168.0.1:80`**. Cisco routers and switches use a proprietary Cisco Discovery Protocol (CDP) to discover information about their neighbors such as model numbers and operating system revision level. However, this is a security weakness as a malicious user could gain the same information. Disable CDP definitely on the boundary router and possibly on the internal routers and switches, dependent upon whether they are required for management software.

Administrative Access

Consider where router access will occur for administration purposes. Security controls enforce policies which determine which interfaces and ports allow an administration connection, and from which network or host will perform the administration; restrict access to those specific locations. Disable unused interfaces and consider static routes to enhance security. Also consider disabling Web-based router configuration. Control physical access to routers.

Do not leave an Internet-facing administration interface available without encryption and countermeasures to prevent hijacking. In addition, apply strong password policies, and use an administration access control system.

Perform router auditing and monitor router logs, and monitor for intrusion detection.

Password Policies

Add a password to the administrator account; many systems are hacked into just because the administrator has left the password blank. Secondly, use complex passwords. Brute force password software can launch more than just dictionary attacks and can discover common passwords where a letter is replaced by a number. Similarly, the Simple Network Management Protocol (SNMP) is probably required for management purposes; although SNMP security is not at all strong, do add passwords (community string) when configuring it. SNMP v3 provides much improved security. Use an administration access control system rather than embedding the administrator's name in the configuration.

Unused Interfaces

Only required interfaces should be enabled on the router. An unused interface is not monitored or controlled, and it is probably not updated. This might expose the Node to unknown attacks on those interfaces. Usually the Telecommunications network (Telnet) protocol is used for administrative access so limit the number of Telnet sessions available and use a time-out to ensure that the session closes if unused for a set time.

Static Routes

Static routes prevent specially formed packets from changing routing tables on the Node's router(s). An attacker might try to change routes by simulating a routing protocol message to cause denial of service or to forward requests to a rogue server. By using static routes, an administrative interface must first be compromised to make routing changes. However, remember that static routes are static; if a link fails the routers will not switch over automatically to use an alternate route, and static routes may need complex configuration.

Web-Based Configuration

If an inbuilt Web server is an optional method for configuration access, as well as a command line mode, disable the Web service as it is probably prone to many **TCP/IP** security weaknesses.

Services

On a deployed router, every open port is associated with a listening service. To reduce the attack potential, default services that are not required should be shut down. Examples include the Bootstrap Protocol (bootps) and Finger, which are rarely required. Enterprise security tools and personnel should also scan the routers to detect which ports are open.

Intrusion Detection

With restrictions in place at the router to prevent TCP/IP attacks, the router should be able to identify when an attack is taking place and notify a system administrator of the attack. Attackers learn what the Node's security priorities are and attempt to work around them. An **intrusion detection system** (**IDS**) can show where the perpetrator is attempting attacks.

Physical Access

Most routers are vulnerable if the attacker can get physical access to the device since they usually have a back-door access method to overwrite the existing configuration so lock the routers away in a room with restricted access.

Switch Security Considerations

There are many things to consider when determining how to secure a switch or other type of link-local network device. As in network devices like routers, they support the appropriate placement of security sensors and controls

to harden the various local area transport boundaries. They also may require associated enterprise security components to manage the policies so deployed and enforced.

Patches and Updates

Install and test patches and updates as soon as they are available on identical hardware and software located in a testing environment. If possible, include real data that has been "sanitized" in the data stores of any system selected for patching, testing or testing patches. For example, a copy of a real DB maybe used, with all sensitive information stripped from it.

VLAN Boundaries

Virtual local area networks (VLANs) allow Node architects to separate network segments and apply access control based on security rules. A VLAN without ACLs provides a first level of security, limiting access to members of the same VLAN. However inter-VLAN traffic is usually required and this is provided by the router routing traffic between the IP subnets and this can be controlled by the use of ACLs. ACLs between VLANs restrict the flow of traffic between different segments of the network. This filtering is typically a simple static packet filter, as opposed to stateful packet inspection or application-layer proxying, which many dedicated firewall devices perform. Using ACLs between VLANs provides an intermediate level of protection by blocking internal intrusions from within the enterprise while intrusions from outside are already blocked by the boundary network. In addition to firewall filtering, VLAN ACLs can also be implemented for an additional layer of security. The disadvantage of implementing ACLs on the VLANs is that they may have an impact on performance and must be configured correctly and efficiently.

Administration Access

Consider where the switch access for administration purposes will occur. Security controls enforce policies which determine which interfaces and ports an administration connection is allowed into, and from which network or host the administration is to be performed. Restrict access to those specific locations. Disable unused interface, and consider static routes to enhance security. Consider disabling Web-based router configuration. In addition, control physical access to routers.

Do not leave an Internet-facing administration interface available without encryption and countermeasures to prevent hijacking. In addition, apply strong password policies, and use an administration access control system.

Perform security auditing, monitor router logs, and monitor for intrusion detection.

Unused Ports

Disable unused Ethernet ports on switches to prevent an unauthorized person with physical access from plugging into an unused port.

Services

Make sure that all unused services are disabled. Also disable Trivial File Transfer Protocol (TFTP), remove Internet-facing administration points, and configure ACLs to limit administrative access.

Encryption

Although not traditionally implemented at the switch, data encryption over the wire ensures that sniffed packets are useless in cases where a monitor is placed on the same switched segment or where the switch is compromised, allowing sniffing across segments.

Internet Boundaries: Subnets

Many administrators use the natural 8-bit boundary in the 16 bits of a class B host ID as the subnet boundary. Subnetting hides the details of internal network organization to external users. Subnets without additional security controls to restrict access are not a good security preventative measure, however simple subnets enable logical and guidance-mandated placement of such controls and will help better manage network performance.

Trusted Guards

Trusted guards are accredited to pass information between two networks at different security levels, such as between SECRET General Service (GENSER) and TOP SECRET Sensitive Compartmented Information (TS SCI), according to well defined rules and other controls.

For additional information see the Trusted Guards [P1150] perspective.

Demilitarized Zone (DMZ)

In computer security a DMZ, based on military usage of the term but more appropriately known as a demarcation zone or perimeter network, is a physical or logical sub network that contains and exposes an organization's external services to a larger, untrusted network, usually the Internet. The purpose of a DMZ is to add an additional layer of security to an organization's LAN, VLAN or subnet; an external attacker only has access to equipment in the DMZ, rather than the whole network.

Firewalls

Firewalls are a form of security sensor and access control package that are embedded at network boundaries between Nodes or between a Node and the larger **Global Information Grid** (**GIG**). They harden the boundaries of and protect the transport network architecture construct known as the **intranet**. Without firewalls, an intranet is only a performance-based grouping of local subnets linked by routers and switches.

Restrict Internet Access to Authorized Sources

Only allow source addresses from the IP network numbers assigned to trusted segments behind the Node's firewall(s), including DMZ networks. This includes primary and secondary network numbers, and subnets that are routed to the Internet through the Node's firewall (including addresses reserved for VPN clients). Apply appropriate subnet masks to trusted networks, i.e., masks that are sufficiently long to identify only that fragment of the IP network number used by Node traffic. For example, if the Node architecture specifies the use of an IETF RFC 1918 (*Address Allocation for Private Internets*) private address from the Class B number **`172.16.0.0`**, and policy only assigns numbers from **`172.16.1.x`**, the configurations should use **`255.255.255.0`** (or **`/24`**), not **`255.255.0.0`** (or **`/16`**) as the subnet mask. Block broadcasts from traversing the firewall's interfaces. While most broadcasts will not pass across **LAN** segments, take measures to ensure this is especially true for Internet-bound packets (or packets destined for any untrusted segment). Prevent traffic from any RFC 1918 private addresses from being forwarded over an Internet access circuit. While Internet service providers (ISPs) block incoming traffic containing private addresses, relying on an external ISP to process traffic according to Node-local policy may not ensure enforcement with any accountability. Block outbound traffic from VLAN workgroups or entire network segments that have no business establishing client connections to Internet servers. If the Node has internal servers that have no business establishing client connections to Internet servers, block all outbound traffic from such systems. An example might be an intranet server that relies entirely on internally provided services (**DNS**, mail, time, etc.) and uses no applications that require Internet access.

Restrict Internet-Accessible Services (Destinations)

Allow outbound connections only to those services the Node's security and acceptable use policies allow for client hosts. Wherever possible, only allow clients to access authorized services from authorized servers. Allow access to service ports Node-internal servers must use to operate correctly, and only allow Node-internal servers access to these services. If the Node operates local mail servers, make certain that only these servers establish outbound **Simple Mail Transfer Protocol** (**SMTP**) connections. (If such measures had been practiced, the Sobig worm, which installed its own SMTP mailing engine, would not have spread so rapidly.) If the Node operates an **HTTP** proxy, or a proxy system that performs some form of Web **URL** or content filtering, only allow outbound proxy connections through the Node firewall. If the Node provides DNS internally, or uses a split DNS, use internal servers as forwarders for the Node-internal trusted network, and only allow outbound DNS requests from the Node's DNS servers so configured. Unless the Node's firewall is participating in routing, block routing protocols at the Node firewall. This is important for large enterprises with multiple firewalls and Internet access routers as well as small operational facilities with broadband connections that use a firewall to exchange and negotiate PPP over Ethernet (PPPoE). Allow any authorized services that make use of unique ports for remote desktop, subscription, licensing channels (e.g., GoToMyPC, BackWeb, and Microsoft). Allow access to these services from hosts that are authorized to use them. Certain network and security vendors use

unique ports for proprietary (and secure) management access. Permit these, but only from hosts used by the administrators of such equipment.

Follow the guidance provided in the STIG for **Domain Name System** (DNS) implementations.

Overlay Network Boundaries

Common examples of overlay network constructs include **virtual private networks** (**VPNs**), and content-based networks (including the localized ones known as DMZs) based on port and protocol firewalls or deep-inspection guards. For further details on subnets and VPNs see the Subnets and Overlay Networks [P1351] perspective.

Performance VPN Access Control

Use a hardened virtual private network (VPN) server to allocate IP address leases and Multi-Protocol Label Switching (MPLS) labels to remote access clients. Use strong authentication to VPN servers.

Protection VPNs

Do not use pre-shared keys. Pre-shared key authentication is a relatively weak authentication method. In addition, pre-shared keys are stored in plaintext. Pre-shared key authentication often is provided for interoperability purposes and to adhere to IP Security (IPsec) standards.

Use the advanced encryption standard (AES) for stronger encryption.

For computers connected to the Internet, do not send the name of the **Certificate Authority** (**CA**) with certificate requests. When using certificate authentication to establish trust between IPsec peers, each IPsec peer sends to the other peer a list of trusted root CAs from which it accepts a certificate for authentication. Each of these CA names is sent as a certificate request payload (CRP), and it must be sent before trust is established. Although transmitting this list aids in connectivity by facilitating the selection of a CA, it can expose sensitive information about the trust relationships of a computer, such as the name of the company that owns the computer and the domain membership of the computer (if an internal public key infrastructure is being used), to an attacker. Therefore, to secure computers that are connected to the Internet, enable the option to exclude the CA name from the certificate request.

For computers connected to the Internet, do not use Kerberos as an authentication method. When using Kerberos V5 authentication during main mode negotiation, each IPsec peer sends its computer identity in unencrypted format to the other peer. The computer identity is unencrypted until encryption of the entire identity payload takes place during the authentication phase of the main mode negotiation. An attacker can send an Internet Key Exchange (IKE) packet that causes the responding IPsec peer to expose its computer identity and domain membership. Use certificate authentication to secure computers that are connected to the Internet.

Do not allow unsecured communication for computers connected to the Internet. If it is Node policy to configure a filter action to negotiate Internet Protocol Security (IPsec), ensure that the following options are disabled in order to secure computers that are connected to the Internet:

- ***Accept unsecured communication, but always respond using IPsec.*** This option allows initial incoming unsecured traffic (for example, TCP SYN packets) but requires protection of outgoing traffic. Disable this option to prevent denial-of-service attacks.
- ***Allow unsecured communication with non-IPsec-aware computers.*** This option allows unsecured communications with computers that cannot negotiate the use of IPsec or process IPsec-secured communications; it is appropriate only in environments where IPsec-secured communication is not necessary.

Tactical and Other Non-IP Networks

Gateways and/or edge routers handle tactical data link local networks such as Link 16. As such they are sub-nets or overlay nets from the wider GIG point of view. Link local networks may require additional boundary protection such as **High Assurance Internet Protocol Encryption** (**HAIPE**), spread spectrum, etc. For further information see the Subnets and Overlay Networks [P1351], Black Core [P1152] and Design Tenet: Encryption and HAIPE [P1247] perspectives.

Content Proxy Networks

Use Domain Name System Security Extensions (DNSSEC) or equivalent directory services to define content routing topologies (Refer to IETF RFC 4033). Use strong authentication with and between proxy servers and message routers.

Use secure directory services such as StartTLS or SLDAP to define **Enterprise Service Bus** (**ESB**) routing topologies.

Overlay Firewalls

Use "black boxes" (like a Nokia IP2255 appliance running Check Point NG) or stripped and hardened dedicated computers as overlay firewalls. The latter choice could involve significantly more maintenance.

Overlay DMZ and Quarantine Zones

Deploy anti-virus gateways at Node network boundaries. In addition, deploy intrusion detection system (IDS), intrusion prevention system (IPS) and other security technologies on at least all outward facing gateways. Nodes should employee virus protection, enabled for both outbound and inbound traffic, at the gateways.

Other Security Concepts

Common DoD-required Transport security controls include the following.

Host, Application, and Network Based IDS/IPS

An intrusion prevention system is a computer security device (generally a software agent, but can be hardware based as well) that monitors network and/or system activities for malicious or unwanted behavior. It can react, in real-time, to block, prevent and or report those activities. The primary difference between an IDS and an IPS system is that IDS only reports where the IPS can take an active role in prevention as well as reporting the activity. The three generally accepted types of IDS/IPS agents are at the network, the operating system, and the application. They perform in one of several ways, like antivirus applications they can use a signature-based, anomaly-based, or hybrid mode to compare observed activity against behaviors that are indicative of potentially malicious outcomes.

Parity Checking

Beyond the standard use of parity checking performed with memory or communications there are also applications that make use of parity checking for the whole computer system. This process could be coded into Node proprietary software, into a Statement of Work (SOW) or Request for Comment (RFC), etc.

Quarantine Concepts and Context

In-Node Transport quarantines are often bundled with the security sensor and controls used to create the boundaries of network constructs such as a DMZ.

Quarantine Zone in DMZ

Most security professionals recognize that a good standard security practice is to implement a quarantine zone within or parallel to the primary DMZ. The main purpose of this is to verify specific installation, configuration and overall compliance with security policy mandates.

Highly Availabilty

Highly available networks require a combination of highly available hardware and software components and highly available distributed components such as routing topologies.

Fault Tolerant and Redundant Networks

Networks are critical Node infrastructure components whose high availability ensure the continuity of net-centric operation. High availability network systems start with the hardware components. If a primary router

fails, traffic may either be switched to an alternate "blade" or be rerouted through alternate network links without any action required on the part of other components.

Multi-Homed Hosts

Nodes should employ network multi-homing to enabling components to connect through alternate networks and not just relying a single network connection whenever mission critical resources, components, or services are not local or organic. Generally, a router or gateway on the external boundary of the Node can accomplish this; multi-homing requires assigning as many network addresses as there are networks employed, requiring management considerations.

Management

Capabilities necessary to Transport network management for enterprise security purposes include the usual two techniques and Component technologies:

Key Management

Refer to IETF RFC 4962, *Guidance for Authorization, Authentication and Accounting Key Management*, for information on network key management.

Auditing and Logging

Most routers have a logging facility and can log all deny actions which would show intrusion attempts. Modern routers have an array of logging features that include the ability to set severities based on the data logged. An auditing schedule should be established to routinely inspect logs for signs of intrusion and probing.

Guidance

- G1667: Implement **Virtual Private Networks** (**VPNs**) in accordance with the guidance provided in the Network **Security Technical Implementation Guide** (**STIG**).
- G1352: Use database clustering and redundant array of independent disks (RAID) for high availability of data.

P1337: User Environment Integrity

User environment boundaries and infrastructure constructs considered separately from the computing infrastructure only emerged with the rise of the **Internet**, the **World Wide Web**, net-centric operations and **service-oriented architectures**. These constructs and boundaries start with physical hardware; software and virtual constructs and boundaries are layered on top. Some of the more established user environment infrastructure constructs include displays and input devices (both real and virtual), client applications, Web browsers and, more recently, rendering engines.

Determining user environment boundaries tends to focus on those subsets of the computing infrastructure resources delegated to and dedicated to a particular user, service agent or process display.

Browser Hardening

Browser hardening is the process of identifying an acceptable Web enabled browser that will function properly with the necessary site accesses. Properly configure the browser to work with the antivirus, antiphishing, antispyware, and firewall solutions. Only download and install a browser from a trusted site and ensure that the digital hashes match before installation. Never run the browser as a "root" or "admin" user.

There are numerous browser **Information Assurance** (**IA**) plug-ins for application, data and services security. Users should either not be able to install additional plug-ins and controls or at least be restricted to approved and **PKI** digitally signed plug-ins and controls. Enable only the those plug-ins and controls that are really needed by the end users, such as Active X, Java controls, etc. Configure these mobile code controls per the DoD Mobile Code policy; see the Mobile Code [P1314] perspective for more information.

Mobile Device Protection

Adopt a multi-tier security approach to mobile security. Set policies to password-protect hand-helds, ensuring employees use strong passwords and personal identification numbers (PINs), and change them frequently to make it difficult for thieves to access confidential information. Protect mobile devices, boundary devices, with internal antivirus gateways, firewall, anti-SMS spam filters, and data encryption technologies. Install regular security updates to protect phones and corporate information from viruses and other malware. Organizations should provide this technology to their employees and teach them how to use it properly. Disable Bluetooth and wireless signals when they are not in use. Bluetooth headsets should be paired exclusively with one employee's handheld device. Regularly scan mobile devices and their information for viruses and other malware. Regularly scan mobile devices and their information for viruses and other malware. Many mobile devices have the capability to receive a "Self Destruct" order which scrambles the internal workings of the device (memory, flash BIOS, etc). This should be a consideration during acquisition and included in concepts of operations (CONOPS) and training.

High Availability Guidance

Employees should schedule regular backups for hand-helds just as they would for any other computer system.

P1338: Data, Application and Service Integrity

Data, application and service boundaries and constructs are virtual; they cannot be separated fully from the underlying computing and transport infrastructures. Generally, they sub-divide these infrastructures in order to prevent interference between, and maintain the integrity of, different mission or business operations. Although the actual boundaries and constructs are operational-specific and consequently a local matter, many of the techniques and technologies used are standard.

Boundary Creation

Formal boundaries in data, applications or services are generally created by application-layer interfaces. Examples include data models and schema, application programming interfaces (APIs) input and output argument datatypes and service protocol interfaces. Baseline hardening such boundaries through type- and range-checking or protocol error handling is a generally standard engineering practice.

Digital Signing

Digital Rights Management (DRM) signing (application) depends on a Trusted Platform Module (TPM) which is used with various operating systems and applications like Windows Vista, Windows Server 2008, and the Linux kernel 2.6.12 and later. It supports capabilities such as Windows BitLocker full-drive encryption technology as well as DRM and software licenses. A TPM microchip is embedded on the computer (or other device) motherboard and stores unique system identifiers along with the decryption keys.

Parity Checking

Beyond assuring integrity by parity checking data in memory or in communications, there are also utilities that make use of parity checking at the services or application level, enabling the "white-listing" of components for execution. White-listing components may more efficiently protect by detecting and preventing zero day exploits, unauthorized software installations, etc. Providing such a capability is a combination of concept of operations (CONOPS) and helper utilities (such as Parity from Bit9 or variant on the open source Tripwire such as Tripwire Enterprise from Tripwire Incorporated).

High Availability

Ensuring high availability of data, applications and services generally is the responsibility of the underlying functional environment infrastructure and not a separate capability. For example, see the *High Availability* subsection in the Computing Infrastructure Integrity [P1335] perspective.

Management

Managing data security, application or service-level security is generally the responsibility of the underlying functional environment infrastructure and is not a separate capability.

Guidance

- G1302: Validate all inputs.

P1178: Identity Management

Identity Management covers the spectrum of tools and processes that serve to represent and administer digital identities and manage access for those identities. Identity is an essential part of the **Core Enterprise Services** (**CES**) Security Services, but CES Increment 1 does not address Identity Management. Identities of **Global Information Grid** (**GIG**) entities, human and non-human (i.e., **services**), must be unique across the GIG. DoD **PKI** X.509 **certificates** reserve a field to contain identity data, but there are issues today with how that field is populated for certain types of users (e.g., coalition partners), and how to handle non-person entities.

While a universal solution for Identity Management is not yet defined, it is possible to make progress in the implementation of these services, particularly for Web applications and services with U.S. users having a **Common Access Card** (**CAC**) holding DoD PKI X.509 certificates.

Identity is not as well understood and defined for non-person entities, such as services that may be part of a long invocation chain that in turn is part of a workflow or is orchestrated to yield a specific answer to a service invocation. The definition of Web server credentialing, though, relies on the DNS name of the site for identification.

The **Net-Centric Enterprise Services** (**NCES**) and **Public Key Infrastructure** (PKI) Program Offices are working on the challenges of non-person Identity Management, and there is a request for information (RFI) to identify potential solutions.

Each identity credential technology varies in strength. The weakest methods are password-based and the strongest are combinations of biometrics and smart cards.

There are also differing strengths within each method. For instance, systems that require complex passwords are stronger than those that accept simple ones, and systems using retina or fingerprint readers are stronger than those that use finger length.

Components that are separate from the implementation of mission- or business-specific functionality often provide identity authentication management and authorization.

Detailed Perspective

- Public Key Infrastructure [P1179]

Guidance

- G1652: Use DoD **PKI** X.509 **certificates** for **servers**.

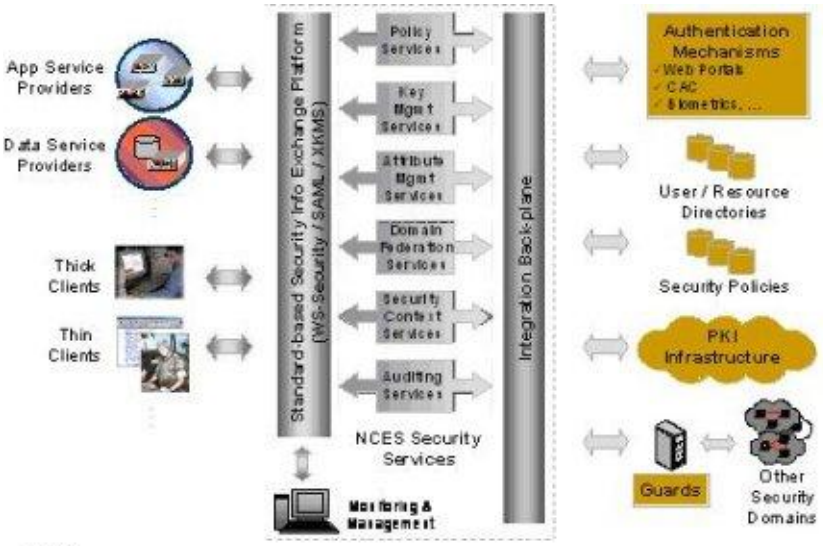
P1179: Public Key Infrastructure

Net-Centric Enterprise Services (**NCES**) Security Services rely heavily on **Public Key Infrastructure** (**PKI**) and **Public Key** (**PK**) Enabling (**PK-Enabling**). PKI provides an assured way for enabled applications to authenticate both intra-node and inter-node. PKI supports the concept of a single login across the **enterprise**, but legacy non-PK-enabled applications and services mean that username and password synchronization is also needed to support the single login concept; however, this is only practical in a limited sense (i.e., not the entire **Global Information Grid** or **GIG**). There remain some PKI implementation challenges, such as the implementation of the process for validating that an entity's **certificate** has not been revoked. Some commercial (**COTS**) products, including some Web Application Containers, do not support the use of the **Online Certificate Status Protocol** (**OCSP**) or do not provide a capability to do file-based checking of the older **Certificate Revocation List** (**CRL**). The U.S. Department of Defense, through the DISA NCES program, supplies Robust Certificate Validation System (RCVS) services for PKI certificates, including **Common Access Card** (**CAC**) credentials; for smart card reader information, see the Common Access Card (CAC) Reader [P1156] perspective. PKI certificate checking includes using OCSP and CRL; the **Joint Interoperability Test Command** (**JITC**) OCSP portal contains more detailed information. For additional PKI-information see the Technologies and Standards for Implementing Software Security [P1391]-related perspectives including Public Key Infrastructure (PKI) and PK Enable Applications [P1061], Key Management [P1041], Certificate Processing [P1009], Encryption Services [P1020], and Smart Card Logon [P1315].

Nodes having both DoD and **Intelligence Community** (**IC**) systems and networks will also face the fact that the DoD and IC have implemented separate PKIs (including the dependent Directory Services). In general, the DoD PKI operates on the collateral classification networks, and the IC PKI operates on classified **Sensitive Compartmented Information** (**SCI**) networks. Nodes may have to interface with multiple PKIs, therefore, depending on the systems and security levels at the Node. This presents some additional challenges when cross-domain interoperation is required, whether intra- or inter-node.

Nodes that have multinational or coalition personnel accessing the system will also encounter a challenge in obtaining CACs containing PKI certificates for these persons. The process is not well defined. As DoD moves further into the net-centric concepts, obtaining certificates for non-human entities in multinational or coalition systems will also be a challenge.

Authorization based on **attributes** corresponding to an entity is a practical way to implement authorization, provided that the enterprise can agree on the definitions of the attributes, policy, and a way of securely communicating and validating role membership. Unfortunately, attribute definitions and common security policy are not defined yet for the **Global Information Grid** (**GIG**), and Nodes are forced to use interim approaches, such as Windows **Active Directory** (**AD**) or **Node Information Services** (**NIS**) group memberships, and evolve to a uniform definition of GIG roles and policies. Federation has not been addressed sufficiently to provide specific guidance.



I1191

Guidance

- G1306: **Identify** and **authenticate** users of the application.

P1339: Authorization and Access Control

Authentication and **identity management** are prerequisites for **authorization** and **access control**. Where authentication and identity management serve to determine "who" (i.e., person or machine) a subject is, authorization and access control determine what privileges a given subject (once identified or authenticated) is allowed for a given resource. In other words, authorization determines what a subject can do with a given resource.

Authorization may grant or deny privileges for resources based on a wider variety of criteria beyond the identity of a subject. Authorization may determine privileges by conditions which may or may not have anything to do with the attributes of the particular subject. For example, user and security roles, the time of day, and location may all be used along with or without the identity of a subject to make a determination for granting privileges.

Because authentication, authorization, and access control are so closely related in most real applications, it is often difficult to discuss them separately. Authentication only establishes the validity of a human or machine entity. Authorization establishes the privileges and span of control for entities, but checking those privileges may be a side effect of being allowed network or physical access rather than checking specific privileges. Access control implements explicit authorization as a combination of policy management components and embedded security control components such as Policy Decision Points (PDPs) and Policy Enforcement Points (PEPs) such as Access Control Sets (ACSs).

The following example is to clarify authorization and access control. Modern files systems are an implementation of authorization and access control. File and directory authorization grants privileges (such as read, write, or execute) to the subject which owns a given file or directory. Additionally, access control is based on the group(s) a subject belongs to in order to grant additional privileges to the subject for the use of a given file or directory.

Various techniques such as roles or attributes may be the basis for access control. (**RBAC**) and Attribute-Based Access Control (ABAC) are examples. For further information on authentication processes see the Design Tenet: Identity Management, Authentication, and Privileges [P1243] perspective. Role definitions are typically within a system boundary and occasionally within or between enclaves. Access control and security often use roles.

Doctrinal spans of control interacting with technical spans of control define net-centric boundaries within Nodes. The presumption in net-centric operations is that the infrastructure extends the span of control beyond the local system; therefore, the limits of the Node technologies define the boundaries.

Authorization policies, therefore, apply within a system and within a Node. Interoperability between Nodes or between a Node and other **Global Information Grid** (**GIG**) systems require federated authorization and protocol negotiations (such as **PKI Certificate Authority** chains and **SAML** transitive trust). In addition, policy may also need alignment through manual negotiation and coordinated configuration.

Restrict the use of administrative credentials in an organization. Administrators can view and modify the security policy settings on computers, network devices, user environments, etc. For this reason, and as a general security best practice, apply the Principle of Least Privilege [P1317] (see *Part 5: Developer Guidance*) throughout the Node.

Authorization and computing infrastructure access control occur at the following main standardized technical boundaries identified by process and storage identifiers: the local system; any virtual machine (VM); any cluster, grid and network file system; and any GIG utility computing grid or network file system.

Authorization and user environment access control occur at the following main standardized technical boundaries or user environment identifiers: the local user account, any virtual machine or browser sandbox.

Process logic access control, such as around **BPEL**, and service access control are generally dependent on security controls within **Web service** infrastructure boundaries. WS-Policy and SAML use XML boundaries, which generally map to data structures and process objects.

Authorization and access control can extend to the transport layer. Use features intended to ensure that a third party cannot intercept, read or alter data transmitted over a network.

For example, SSL allows for authenticating and controlling access to data over an **HTTP** connection using **credentials** (such as a client or server digital **certificate**). Access may be controlled for a given subject (such as a user or client system) or a group of subjects (for example all users belonging to a given certificate authority).

With SSL communication, any of the following authentication scenarios are possible:

- No SSL authentication (or null authentication): The server does not send a certificate and does not request a certificate from the client. From an SSL perspective, the server does not know who the remote client is, or accepts any certificate that the client may present.
- One-way SSL authentication: Either the server or the client, but not both, requires certificates. Server authentication, for example, is one-way authentication where the server sends its certificate to the client but does not request a certificate from the client. Alternatively, the server may require a certificate, but does not send one and the client does not require one.
- Two-way SSL authentication: This is client and server authentication, where the server sends a certificate required by the client and also requires the client to send a certificate.

Configuring SSL authentication in the server is independent of configuring SSL authentication in the client.

Guidance

- G1306: **Identify** and **authenticate** users of the application.

P1340: Confidentiality

Confidentiality is the property of preventing disclosure of information to unauthorized individuals or systems. For example, a credit card transaction on the Internet requires transmitting the credit card number from the buyer to the merchant and from the merchant to a transaction processing network. The system attempts to enforce confidentiality by encrypting the card number during transmission, by limiting the places where it might appear (in databases, log files, backups, printed receipts, and so on), and by restricting access to the places where it is stored. If an unauthorized party obtains the card number in any way, a breach of confidentiality has occurred.

Breaches of confidentiality take many forms. Permitting someone to look over your shoulder at your computer screen while you have confidential data displayed on it could be a breach of confidentiality. If a laptop computer containing sensitive information about a company's employees is stolen or sold, it could result in a breach of confidentiality. Giving out confidential information over the telephone is a breach of confidentiality if the caller is not authorized to have the information.

Confidentiality is necessary (but not sufficient) for maintaining the privacy of the people whose personal information a system holds. Confidentiality and privacy control occurs in computing, network and user environment infrastructure.

- ***Computing Infrastructure*** confidentiality and privacy control occur within standardized technical boundaries such as the local system; a virtual machine (VM); a Node cluster, grid and network file system; and a **Global Information Grid** (**GIG**) utility computing grid and network file system. This requires protection (usually encryption) of both the virtual storage and virtual network protocols through secure transports.

- ***Network Infrastructure*** confidentiality and privacy control occur within the following standardized technical boundaries by offering either physical protection or payload encryption: the local area subnet or VLAN, the intranet subnets, any relevant overlay networks, and the GIG internet (e.g., **SIPRNet**).

- ***User Environment Infrastructure*** confidentiality and privacy control occur within the following standardized technical boundaries through access control privileges: the local user account and any virtual machine (VM) or browser sandbox.

- ***Data, applications and services*** confidentiality and privacy control occur within the following standardized technical boundaries or application identifiers: the local application or service invocation or session context, Web page context, or application field context.

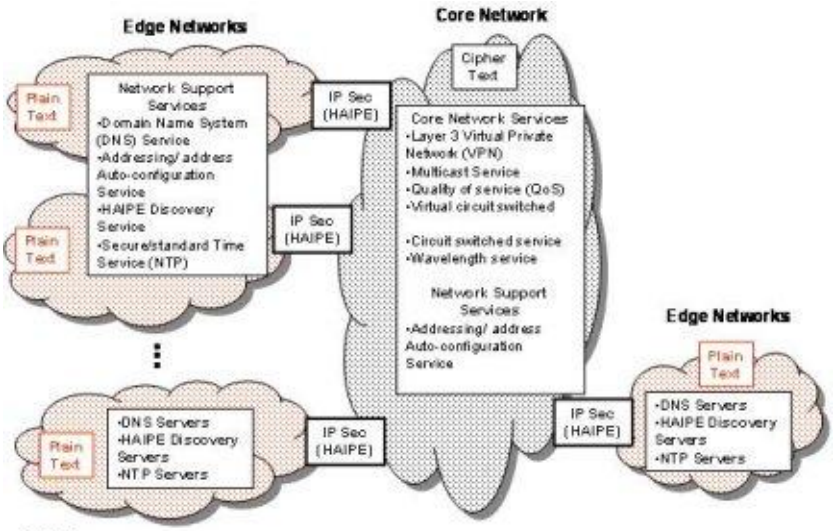
Detailed Perspective

Black Core [P1152]

P1152: Black Core

The DoD will be aggregating **Internet Protocol** (IP) packet traffic from multiple security enclaves onto network segments secured at the network layer in the protocol stacks; these segments, called the Black Core, are enabled through the use of **High Assurance Internet Protocol Encryption** (HAIPE) devices. Challenges to the implementation of HAIPE devices and the Black Core include organic support for the following: IP-based **quality of service** (QoS), dynamic unicast IP routing, support for dynamic **multicast** IP routing, support for mobility, and support for simultaneous **Internet Protocol Version 6** (IPv6) and **Internet Protocol Version 4** (IPv4) operation.

The Black Core is a concept fundamental to **Global Information Grid** (GIG) networking, but actionable guidance is still in its infancy. Interoperability with the Black Core will require active monitoring by the Node's management and program offices. The basic architecture of the Black Core is shown below. The Node typically provides one or more edge networks as shown in the diagram, along with the services indicated. The edge (Node) networks are sometimes referred to as **Plain Text** (PT) networks, while the Black Core is the **Cipher Text** (CT) network.



I1182

Best Practices

- BP1670: Plan for Black Core implementation in the local Node.
- BP1671: Consider Black Core transition whenever there is a significant Node network design or configuration decision to make in an effort to avoid costly downstream changes caused by Black Core transition.

P1150: Trusted Guards

Trusted guards are accredited to pass information between two networks at different security levels, such as between ***SECRET General Service (GENSER)*** and ***TOP SECRET Sensitive Compartmented Information (TS SCI)*** level networks, according to well defined rules and other controls. Guard products only pass defined types of information (e.g., email, images, or formatted messages). A key challenge is how to implement net-centric operations across trusted guards in the presence of **CES** services. See the Cross-Domain Interoperation [P1169] perspective for additional information.

Best Practices

- BP1653: Do not build dedicated Node guard products.
- BP1654: Do not build dedicated **Component** guard products.
- BP1668: Acquire and configure approved guard products with the help of the Government program offices that acquire such guards.
- BP1669: Select **XML**-capable **trusted guards**.

P1372: User Interface Services

This service area relates to how users interact with applications. Use the following detailed perspectives for NESI guidance related to this service area.

Detailed Perspectives

- User Interfaces [P1058]
- Human-Computer Interaction [P1032]

P1058: User Interfaces

The user interface represents all the components used to generate an interactive display that enables users to communicate with applications. The components of a user interface are not necessarily in the same physical location. For example user interface components are found both client side (as in the case of **HTML** pages) and server side (as in the case of components that generate HTML pages).

The following perspectives provide guidance for building user interfaces to promote interoperability of user interface components and improve human-computer interactions.

Detailed Perspectives

- Human-Computer Interaction [P1032]
- Browser-Based Clients [P1008]
- Thick Clients [P1074]

P1032: Human-Computer Interaction

Human-Computer Interaction (HCI) is the study, planning, and design of the interaction between humans and computers. HCI is a subset of Human Systems Integration (HSI). Human Systems Integration is a requirement for **Department of Defense** (**DoD**) acquisition; see as Enclosure 8 of DoD Instruction 5000.02 [R1165]. In particular, this instruction requires that Program Managers shall take steps to include human factors engineering during system engineering over the lifecycle of the program to provide effective human-machine interfaces, "Where practical and cost effective, system designs shall minimize or eliminate system characteristics that require excessive cognitive, physical or sensory shills; entail extensive training or workload-intensive tasks; result in mission-critical errors; or produce safety or health hazards."

Interoperability includes both the technical exchange of information and the end-to-end operational effectiveness of that exchanged information as required for mission accomplishment. Whenever a user is required to interact with a computer user interface to accomplish a mission, and that interaction fails due to poor design (i.e., information is misunderstood or interaction results in a high cognitive load) then the risk of not accomplishing the mission is increased.

This perspective provides guidance and best practices that benefit human computer interaction to increase total system performance, reduce maintenance costs through better design, and accommodate the cognitive characteristics of the user. This perspective provides guidance for human factors common to all applications including data entry, data display, and user control appearance and behavior. The following detailed perspectives provide additional human factor guidance on more specific topics.

Detailed Perspectives

- Human Factor Considerations for Web-Based User Interfaces [P1108]
- Designing User Interfaces for Accessibility [P1111]
- Designing User Interfaces For Internationalization [P1112]

Guidance

- G1760: Solicit feedback from users on user interface usability problems.
- G1761: Provide units of measurements when displaying data.
- G1762: Indicate all simulated data as simulated.
- G1763: Indicate the security classification for all classified data.
- G1032: Validate all input fields.
- G1268: Label all data entry fields.
- G1270: Include scroll bars for text entry areas if the data buffer is greater than the viewable area.
- G1285: Use **relative font sizes**.
- G1286: Provide text labels for all buttons.
- G1287: Provide feedback when a transaction will require the user to wait.

Best Practices

- BP1280: In tabular data displays, right justify integer data.
- BP1281: In tabular data displays, justify numeric data with decimals by using the decimal point.
- BP1767: Follow a standard process for human systems integration engineering such as the one defined by the International Organization for Standardization in ISO 13407:1999 on human-centered design processes for interactive systems.
- BP1272: Disable dependent child controls when the parent control is inactive.
- BP1273: Gray out the push button label if a button is unavailable.
- BP1290: Use a tool tip to display help information about a control when the purpose of the control is not self-evident.

- BP1291: Use obvious navigation controls for moving between pages in search results that span multiple pages.
- BP1298: Provide basic search functionality as the default with a link or button that provides more advanced search features.
- BP1054: Use standard controls that provide input choices for the user.

P1111: Designing User Interfaces for Accessibility

Section 508 of the Rehabilitation Act of 1973, as amended, requires that individuals with disabilities have access to and use of information that is comparable to that provided to federal employees and members of the public who are not disabled. The standards created under Section 508 define technology accessibility requirements for all types of information technology in the federal sector, including Web-based intranet and Internet information and applications.

Federal accessibility standards focus on providing redundancy in information presentation and interaction so individuals with disabilities can use different modalities to access information. The scope of Section 508 is confined to the federal sector, with a limited exemption for systems used for military command, weaponry, intelligence, and cryptologic activities. The exemption does not apply to routine business and administrative systems used for other defense-related purposes or by defense agencies or personnel. A Web application or portal that will be used in these systems is required to comply with Section 508 standards.

Guidance

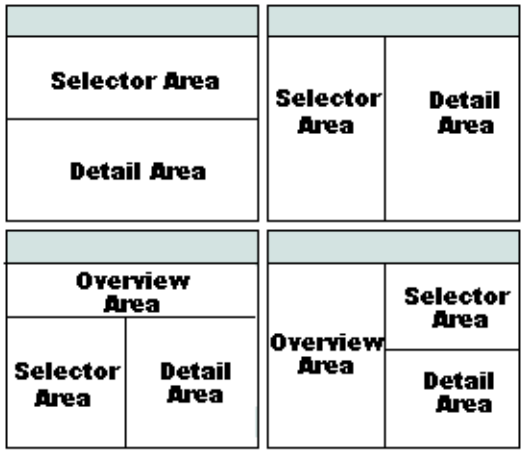
- G1044: Comply with Federal accessibility standards contained in Section 508 of the Rehabilitation Act of 1973 (as amended) when developing software user interfaces.

P1108: Human Factor Considerations for Web-Based User Interfaces

Web based user interfaces include **Web sites**, **Web applications**, and **Web portals**. This perspective provides guidance and best practices relating to human factors consideration that are specific to Web-based user interfaces. Additional information concerning general user interface guidance is available in the Human Computer Interaction [P1032] perspective.

Web sites tend to be content-centric and are generally developed using **HTML** for marking up content for Web pages. Sometimes other technologies such as **JavaScript** are used to add interactivity to Web pages. If developers choose to use a mix of HTML and other technologies to deliver Web content, it is important that they design their Web pages so the pages work correctly when viewed with browsers that support these technologies as well as with browsers that do not. In this way, all users will have an acceptable experience using the Web site.

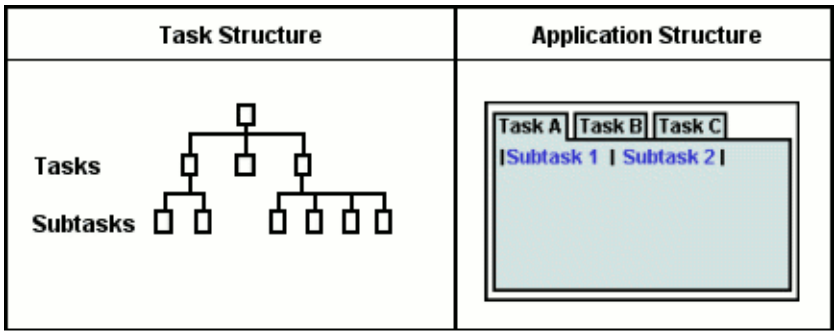
Web sites vary in their layout, but there are common themes for layouts that are widely used and understood users. Some example Web site layouts are shown in this figure:



I1178

Web Applications

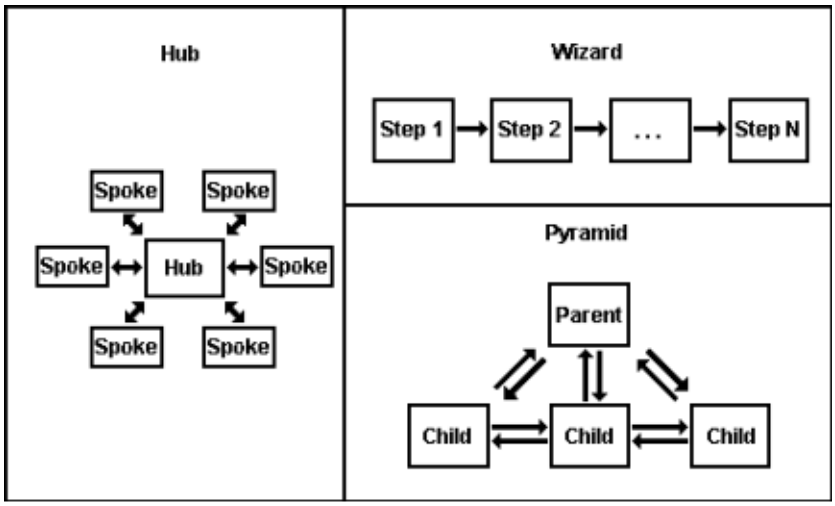
A Web site tends to be content-centric, but a Web application tends to be task-centric and organizes content around a hierarchy of tasks. An example user interface for a given task structure is shown in this figure:



I1179

A Web application often supports interactivity similar to that available in a desktop application but delivered to users within the framework of a browser. Because a Web application allows users to create, save, and delete data, it supports greater complexity in design and interactivity compared to a content-oriented site.

In addition to application structure, there are common navigation models that are well understood by users for Web application workflow. Some common examples are in this figure:



I1180

The "hub navigation metaphor" is often used for applications where a task consists of multiple independent steps that are performed in any order. The hub page present users with a collection of "spoke" pages that they access from a single page; when users submit their input, they are returned to the hub page.

The "wizard navigation" metaphor is often used when a task consists of multiple interdependent steps that are performed in a predefined order. In this metaphor, a wizard presents users with a collection of pages that they interact with sequentially; when the user submits their input, the user is presented with the next page

The "pyramid navigation" metaphor is often used when it is important to navigate to sibling, child, or parent pages while completing tasks; when the user submit their input, they are returned to the same page where they follow links to another adjacent page in the pyramid.

Web Portals

A portal is a type of Web application that provides a gateway from which users can access the information, resources, and services they need. A portal aggregates and organizes content from different sources within a Web page related to specific mission or business task. Sometimes a portal allows users to personalize what and how information is presented to them such as selecting and arranging the content presented on the portal page and to choosing the "look and feel" of the display.

The pages in a portal contain portlets that enable users to view and/or interact with Web-based information related to a specific function. A portlet provides more than a view of existing Web content, functioning instead as a complete application with multiple states and view modes.

Since portals are designed to contain portlets from various sources, it is important for portlet developers to develop portlets carefully to allow for a standard presentation and behavior when the portlet is deployed within the portal. Allowing for configuration for presentation such as fonts and colors allows for a common look and feel across all portlets within a portal. Developing portlets according to standards for user controls enables a better experience for the end user with respect to common portlet control behavior.

Guidance

- G1267: Use industry standard **HTML** data entry fields on **Web pages**.
- G1276: Do not modify the contents of the Web browser's status bar.
- G1277: Do not use tickers on a Web site.
- G1278: Use the browser default setting for links.
- G1284: Use only one font for **HTML** body text.
- G1292: Use text-based Web site navigation.

- G1294: Provide a site map on all Web sites.
- G1295: Provide redundant text links for images within an **HTML** page.
- G1566: Use **`alt`** attributes to provide alternate text for non-text items such as images.
- G1759: Use a style guide when developing Web portlets.

Best Practices

- BP1297: Structure a Web site hierarchy so users can reach important information and/or frequently accessed functions in a maximum of three jumps.
- BP1299: Include a link back to the home page on all Web pages.
- BP1042: Do not build a **Web page** where the horizontal width is greater than the screen (vertical scrolling is fine), planning for the lowest common denominator to be super-VGA resolution (800 x 600).
- BP1041: Do not change the default colors of the links.
- BP1038: Use a **sans serif** font (e.g., Arial, Verdana) in Web pages rather than a serif font (e.g., Times New Roman).
- BP1039: Do not underline any text unless it is a link.
- BP1768: Use design patterns for application navigation.

P1008: Browser-Based Clients

This complex perspective provides guidance for creating and interfacing to thin clients. It includes the following topics:

- XML Rendering [P1084]
- Active Server Pages (ASP) [P1001]
- Active Server Pages for .NET (ASP.NET) [P1002]
- Java Server Pages (JSP) [P1040]
- Style Sheets [P1070]

Guidance

- G1035: Follow W3C standards for code which will generate a Web page display.
- G1043: Separate formatting from data through the use of **style sheets** instead of hard coded **HTML** attributes.
- G1271: Provide instructions and **HTML** examples for all style sheets.
- G1283: Use **linked style sheets** rather than embedded styles.

Best Practices

- BP1040: Use hex codes for all colors (e.g., #FFFF33), never the color name (e.g., yellow).
- BP1291: Use obvious navigation controls for moving between pages in search results that span multiple pages.
- BP1567: Use the <**abbr**> and <**acronym**> tags to specify the expansion of acronyms and abbreviations.
- BP1568: Use a markup language to represent mathematical equations within Web pages.

P1084: XML Rendering

XML can render display-device-neutral output to a particular output device given a set of display rules or a **style sheet**. The **XSLT** file is the decoupled output formatter that determines how the output device renders the data.

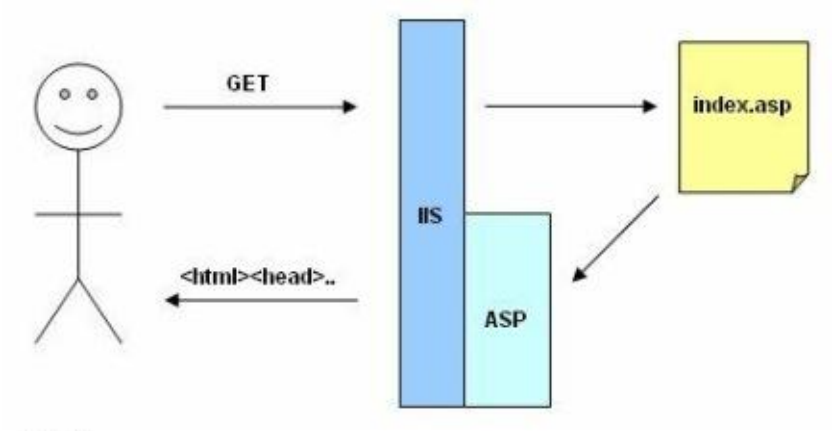
Guidance

- G1045: Define **XML** format information separately in **XSL**.

P1001: Active Server Pages (ASP)

Active Server Pages (**ASP**) are scripts that are executed by Microsoft **Internet Information Services** (**IIS**). The output is returned to the **end user** as **HTML**. Typically, an ASP script generates a customized **Web page** on the fly before sending it to the end user.

- **Active Server Pages**:
  - Are specific to Microsoft
  - Only run on Internet Information Services (IIS) or Personal Web Server (PWS).
  - Can contain HTML, **Jscript**, and VBScript
  - Can access **Component Object Model** (**COM**) component



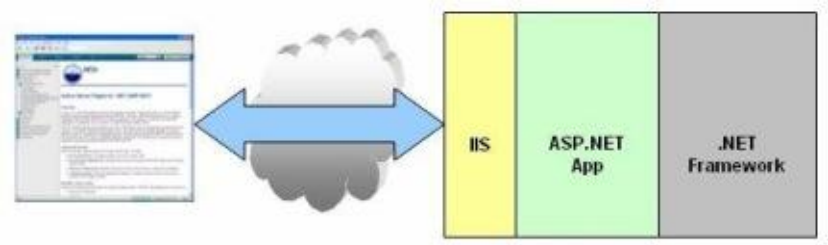
I1027

Guidance

- G1050: In **ASP**, isolate the presentation tier from the middle tier using **COM** objects.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.

P1002: Active Server Pages for .NET (ASP.NET)

Microsoft .NET uses ASP.NET for Web applications. ASP.NET requires Microsoft **Internet Information Services** (**IIS**).



I1029

ASP.NET improves upon ASP. It has more features than **Java Server Page** (**JSP**), an extensible Web technology that uses static data, **JSP** elements, and server-side Java objects to generate dynamic content for a client. Typically, the static data are **HTML** or XML elements, and in many cases the client is a Web browser. An application responds to events, such as code-behind and event-driven Web controls.

Guidance

- G1052: Use the code-behind feature in ASP.NET to separate presentation code from the business logic.
- G1053: Do not embed HTML code in any code-behind code used by aspx pages.
- G1056: Specify a versioning policy for **.NET** assemblies.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.

P1040: Java Server Pages (JSP)

Java Server Page (**JSP**) technology enables Web developers and designers to develop and maintain information-rich, **dynamic Web pages** that leverage existing business systems rapidly and easily. As part of the Java technology family, JSP technology enables rapid development of platform-independent, Web-based applications. JSP technology separates the user interface from content generation, enabling designers to change the overall page layout without altering the underlying dynamic content.

Java Server Pages:

- Are similar to **ASPs**.
- Can contain **HTML**, Java code, and JavaBean components
- Provide a powerful, **dynamic Web page** assembly mechanism
- Are platform-independent
- Are compiled into Servlets at runtime; on most application servers, this occurs only the first time they are invoked

Guidance

- G1060: Encapsulate Java code that is used in **JSP**(s) in tag libraries.
- G1058: Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.

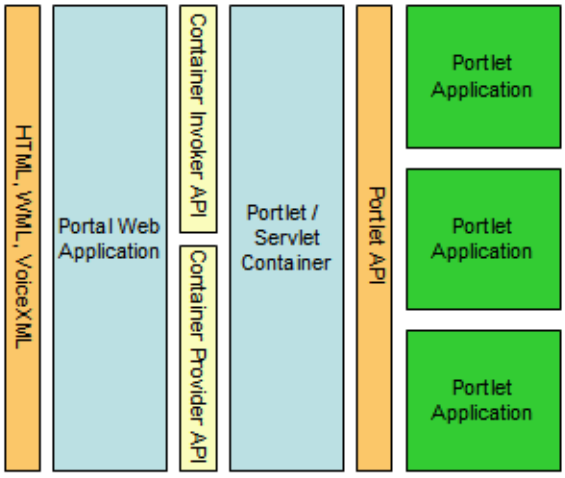
P1077: Web Portals

A **Web portal** is a **Web site** that provides a starting point or gateway to other resources on the Internet or an intranet. Access to a Web portal is typically via **HTTP** and can be in any number of formats including **HTML**, **Wireless Markup Language** (**WML**) or **VoiceXML**. A Web Portal often uses a **Web Application** that provides **single sign-on**, content **integration** and **aggregation** from different sources, **collaboration**, content and document management and personalization of the presentation. It hosts the presentation layer of different backend systems in a **single touch point**.

An attractive feature of a **portal** to an **enterprise** is to aggregate different applications into a single **page** with a common **Look and Feel** that enhances the portal **end user's** experience. A portal may also have sophisticated personalization features, which provide customized content to individual end users or to their roles within the enterprise. **Portal pages** can dynamically coordinate different **portlets** to create specialized content for different portal end users.

IBM's Websphere depicts the basic architecture of portals as a series of layers between the end user's environment such as **browsers**, mobile devices and phones. The portal processes an end user **client** request. A Web Application that interacts with the portlet to request the web page for the current end user is produced. The portal Web Application then uses the **portlet container** for each portlet to retrieve the requested content through the **Web Container Invoker API**. The portlet container calls the portlets through the Portlet API. The Container Provider Service Provider Interface (SPI) enables the Web Application to retrieve information from the portal through its portlet container.

The portlet container invokes the portlets, provides a runtime environment, and manages the lifecycle of the portlet. In addition, it provides persistence for the portlet to store end user information enabling the production of customized Web pages.



I1006

Guidance

- G1245: Isolate the **Web service portlet** from platform dependencies using the **Web Services for Remote Portlets** (**WSRP**) Specification protocol.

Best Practices

- BP1246: Base Java-based portlets on **JSR 168**.
- BP1247: Encapsulate Java-based **portlets** in a **.war** file.

P1070: Style Sheets

A **style sheet** is a template used to customize the layout of a **Web site**. Style sheets allow Web sites to present content in a consistent manner. Web designers can create custom tags to override default values:

```
h1,h2,h3 {
   font-family: verdana, arial, 'sans serif';
}
p,table,li {
   font-family: verdana, arial, 'sans serif';
   margin-left: 10pt;
}
```

Guidance

- G1043: Separate formatting from data through the use of **style sheets** instead of hard coded **HTML** attributes.
- G1283: Use **linked style sheets** rather than embedded styles.
- G1271: Provide instructions and **HTML** examples for all style sheets.

Best Practices

- BP1040: Use hex codes for all colors (e.g., #FFFF33), never the color name (e.g., yellow).
- BP1041: Do not change the default colors of the links.
- BP1038: Use a **sans serif** font (e.g., Arial, Verdana) in Web pages rather than a serif font (e.g., Times New Roman).

P1074: Thick Clients

A thick client (often called "fat client") is a client machine in a client/server environment that performs most or all of the application processing with little or none performed in the server.

Guidance

- G1030: Use a user interface **component** library.

P1373: User (Physical/Cognitive)

The service area relates to the security protection of the system in an external environment also defined as the cognitive aspect of the Human Machine Interface (HMI). Use the following detailed perspective for NESI guidance related to this service area.

Detailed Perspective

- Human-Computer Interaction [P1032]

P1374: Exposure Verification Tracking Sheets

Chairman Joint Chiefs of Staff Instruction (CJCSI) 6212.01E, *Interoperability And Supportability of Information Technology*, 15 December 2008 [R1175] specifies the use of Exposure Verification Tracking Sheets as part of compliance verification activities regarding the *DoD Net-Central Data Strategy* [R1172] and the *DoD Net-Centric Services Strategy* [R1313].

Completion of Data and Service Exposure Verification Tracking Sheets is a requirement only when one or more of a program's Information Technology (IT) and National Security System (NSS) nodes has identified, within its architecture views, a requirement to use the GIG to transport or store data or information. Completion of Exposure Verification Tracking Sheets is not a requirement for the following systems:

- Programs with only point-to-point or platform-centric information exchanges
- Communication transmission systems
- Tactical systems operating exclusively in a disadvantaged communications environment
- Individual tactical remote sensors
- Systems with legacy waivers

In addition, for joint or multi-Service systems, CJCSI 6212.01E recommends requiring that only the Joint Program Office or lead Service meet the reporting requirement.

An *Exposure Verification Tracking Sheet Guide* dated 27 December 2007 is located on the CJCSI 6212 Resource Page (available at https://www.intelink.gov/wiki/Portal:CJCSI_6212_Resource_Page; access restrictions may apply). This guide aids selecting the type of tracking sheet(s) for each program and provides instructions for the completing the sheets.

Use the definitions for data and services along with the decision flowchart in the Exposure Verification Tracking Sheet Guide to determine if a Data or Service (or both) Exposure Verification Tracking Sheet is required. In addition, the linked detailed perspectives are useful in tracking NESI perspectives and their associated guidance and best practice statements to Data and Services Exposure Verification Tracking Sheets. This set of NESI content traceability, along with an associated NESI checklist, is useful to programs when completing the tracking sheets.

Detailed Perspectives

- Data Exposure Verification Tracking Sheet [P1375]
- Service Exposure Verification Tracking Sheet [P1380]

P1375: Data Exposure Verification Tracking Sheet

The Data Exposure Verification Tracking Sheet asks for the following information (grouped by Joint Capability Area) for each Program or System of Record:

- The name of the data asset as it is registered in the NCES Enterprise Catalog Service
- A short description of the exposed data assets
- A status (Objective Achieved, In Progress, Project at Risk, Progress Stopped, or Exposure not Started) for the following:
  - Visibility - able to be seen, detected, or distinguished and to some extent characterized by humans and/or information technology systems, applications, or other processes
  - Accessibility - measured both in terms of policy and operational considerations of humans, systems, or applications being able to retrieve data within an asset
  - Understandable - capable of being comprehended in terms of subject, specific content, relationships, sources, methods, quality, spatial and temporal dimensions, and other factors

Detailed Perspectives

- Data Visibility [P1376]
- Data Accessibility - Policy [P1377]
- Data Accessibility - Operational [P1378]
- Data Understandability [P1379]

P1376: Data Visibility

Making data assets visible includes creating and associating **metadata** with the data asset. A data asset is visible when discovery metadata that describes the asset is accessible. Use the **DoD Discovery Metadata Specification** (**DDMS**) to specifying discovery metadata. The linked detailed perspectives apply to data visibility.

Detailed Perspectives

- Design Tenet: Make Data Visible [P1250]
- Design Tenet: Provide Data Management [P1257]
- Net-Centric Data Strategy (NCDS) [P1204]
- Metadata Registry [P1050]

P1204: Net-Centric Data Strategy (NCDS)

Information sharing is a core concern of DoD enterprise integration and data is the critical element underlying information sharing. Goals of the DoD Net-Centric Data Strategy (NCDS) [R1172] include making **data** visible, accessible, understandable, and trustable while maintaining security.

DoD Directive 8320.2, *Data Sharing in a Net-Centric Department of Defense*[R1217] contains guidance for the implementation of the **DoD Net-Centric Data Strategy** (**NCDS**) within the Services. It directs the heads of DoD components to establish plans, programs, policies, processes, and procedures to implement the NCDS. The following best practices, adopted from the Electronic Systems Center Net-Centric Data Strategy Implementation Roadmap (reference [R1218]), guide a program's response to the NCDS as part of its net-centric migration.

While the goal of the NCDS is to make all data visible, accessible, and understandable, some data will be more important to share across a broader community than other data. Some data are easier to share than other data. Data can be targeted to be shared within specific communities or it can be made available for general use by unanticipated users. Data can be shared effectively via data access services using SOA. Coordinating data sharing development efforts across multiple programs requires programs to share their data-related development plans.

Identify types of data items for potential sharing external to the program. Potential sources for this information include descriptions of existing data stores and existing or planned interfaces, architectural products, data models, document repositories, etc. Consider the logical entities represented by the data. Consider issues related to security classification, frequency of exchange, and file formats. Consider issues related to timeliness and data quality.

Identify specific data items for potential sharing external to the program. Potential sources for this information include descriptions of existing data stores and existing or planned interfaces, architectural products, data models, document repositories, etc. Identify the source, typical destinations, security classification, frequency of exchange, and typical size of the data. Avoid sharing data from other sources as a "pass through."

Prioritize data items for potential sharing external to the program. Analyze key operational processes to identify operationally important information exchanges. Consult with **Communities of Interest** (**COIs**) to determine the demand for specific data assets. Consider such factors as cost, time, and engineering difficulty.

Publish preliminary program data-related development plans. While initially incomplete, preliminary program data-related development plans may prove useful to other programs as they plan their migrations due to the inherent interdependencies introduced by the Net-Centric Data Strategy. Create initial descriptions of data items that are forecast to be sharable using the **DoD Discovery Metadata Specification** (**DDMS**) and publish them in the **DoD Metadata Registry**.

Create external representations for sharable data items. Coordinate both internally within the program and externally with appropriate COIs. Explore de facto loose coupler and existing COI data formats. Create XML schema definitions for the data items and publish them in the DoD Metadata Registry.

Create **metadata** representations for sharable data items. Identify what data items will be searchable taking into account cost and performance considerations. Tag individual data items as appropriate using automated metadata generation where possible. Use the DDMS to define discovery metadata.

Implement and publish data access services. Select the appropriate underlying SOA-based technologies using NESI. Design service interfaces using the XML schema definition for the data exchange. Take into account security, performance, and versioning considerations. Use DDMS and the DoD Metadata Registry. For **SOAP**-based services, consider DoD efforts related to **WSDL** and **UDDI**-based registries [R1220]. Test, deploy, and sustain data exchange mechanisms that support the NCDS in much the same fashion as any other mission-oriented software. The standard lifecycle methodologies used for other systems and software will apply.

Best Practices

- BP1855: Identify types of data items for potential sharing external to the program.
- BP1856: Identify specific data items for potential sharing external to the program.

- BP1857: Prioritize data items for potential sharing external to the program.
- BP1858: Publish preliminary program data-related development plans.
- BP1859: Create external representations for sharable data items.
- BP1860: Create **metadata** representations for sharable data items.
- BP1861: Publish data access services that implement interfaces to shared data.
- BP1863: Make shareable data assets visible, even if they are not accessible.
- BP1865: Provide sufficient program, project, or initiative **metadata** descriptions and automated support to enable **mediation** and translation of the data between **interfaces**.

P1377: Data Accessibility - Policy

Making data assets accessible includes making the data assets available in shared spaces, providing acceptable boundaries for security along with policies and information in place to govern the access of such data. The linked detailed perspective applies to data accessibility with respect to policy criteria.

Detailed Perspective

- Net-Centric Data Strategy (NCDS) [P1204]

P1378: Data Accessibility - Operational

Making data assets accessible includes making the data assets available in shared spaces, providing acceptable boundaries for security along with policies and information in place to govern the access of such data. The linked detailed perspectives apply to data accessibility with regard to operational criteria.

Detailed Perspectives

- NCES Federated Search [P1182]
- Net-Centric Data Strategy (NCDS) [P1204]

P1379: Data Understandability

Making data assets understandable includes publishing associated semantic and structural **metadata** in the **DoD Metadata Registry**. It also includes using well-defined data elements to establish the semantic basis for data models including database models **XML** data models. The linked detailed perspectives apply to data understandability

Detailed Perspectives

- Net-Centric Data Strategy (NCDS) [P1204]
- Metadata Registry [P1050]
- Design Tenet: Make Data Understandable [P1253]
- Data Modeling [P1003]
- Metadata [P1049]
- XML Semantics [P1096]

P1380: Service Exposure Verification Tracking Sheet

The Service Exposure Verification Tracking Sheet asks fot the following information (grouped by Joint Capability Area) for each Program or System of Record:

- The name of the service as it is registered in the **NCES** Services Registry
- A short description of the service
- The type of the service
- The name of the submission package name for the service as entered into the **DoD Metadata Registry**
- A status (Objective Achieved, In Progress, Project at Risk, Progress Stopped, or Exposure not Started) for:
  - Visibility - measured both in terms of registration and discoverability of services being able to be seen, detected, or distinguished and to some extent characterized by humans and/or information technology systems, applications, or other processes
  - Accessibility - measured both in terms of policy and registration considerations regarding humans, systems, or applications being able to retrieve data within an asset
  - Understandable - measured both in terms of registration and conformance with **community of interest** (**COI**) vocabularies of being capable of comprehension in terms of subject, specific content, relationships, sources, methods, quality, spatial and temporal dimensions, and other factors

Detailed Perspectives

- Service Visibility - Registered [P1381]
- Service Visibility - Discoverable [P1382]
- Service Accessibility - Policy [P1383]
- Service Accessibility - Registered [P1384]
- Service Understandability - Registered [P1385]
- Service Understandability - COI Data Models [P1386]

P1381: Service Visibility - Registered

Making services visible includes registering services in the **enterprise service registry** (i.e., publish the **metadata** describing the services) to ensure that potential users will be able to discover the services. The linked detailed perspectives apply to service visibility with regard to service registration.

Detailed Perspectives

- Service Definition Framework [P1296]
- Service Enablers [P1325]
- Metadata Registry [P1050]
- XML Semantics [P1096]
- WSDL [P1082]

P1382: Service Visibility - Discoverable

Making services visible includes registering services in the **enterprise service registry** (i.e., publish the **metadata** describing the services) to ensure that potential users will be able to discover the services. The linked detailed perspectives apply to service visibility with regard to discoverability.

Detailed Perspectives

- Universal Description, Discovery, and Integration (UDDI [P1075])
- Metadata Registry [P1050]
- WSDL [P1082]
- Service Definition Framework [P1296]
- Service Enablers [P1325]
- XML Semantics [P1096]

P1383: Service Accessibility - Policy

Service accessibility concerns the ability to discover services and access them in a timely, secure, and effective manner. Service accessibility includes security mechanisms that determine access. The linked detailed perspectives apply to service accessibility with regard to policy.

Detailed Perspectives

- Metadata Registry [P1050]
- Design Tenet: Identity Management, Authentication, and Privileges [P1243]

P1384: Service Accessibility - Registered

Service accessibility concerns the ability to discover services and access them in a timely, secure, and effective manner. Service accessibility includes security mechanisms that determine access. The linked detailed perspectives apply to service accessibility with regard to registration.

Detailed Perspectives

- Metadata Registry [P1050]
- Universal Description, Discovery, and Integration (UDDI) [P1075]
- WSDL [P1082]
- Service Definition Framework [P1296]
- Service Enablers [P1325]

P1385: Service Understandability - Registered

In order for services to be understandable, providers of services must use a common set of service description information to enable consistent discovery by users throughout the **enterprise**. The linked detailed perspectives apply to service understandability with regard to registration.

Detailed Perspectives

- Metadata Registry [P1050]
- Metadata [P1049]
- XML Semantics [P1096]

P1386: Service Understandability - COI Data Models

In order for services to be understandable, providers of services must use a common set of data models for data passed to and from services. The linked detailed perspectives apply to service understandability with regard to **community of interest** data models.

Detailed Perspectives

- Metadata Registry [P1050]
- Metadata [P1049]
- XML Semantics [P1096]
- Data Modeling [P1003]

G1001

Use formal standards to define public interfaces.

Rationale:

It is important to use a common language to define the interfaces so producers and consumers can work independently and together.

There are many standards for defining interfaces (**UML**, **WSDL**, and **CORBA**). Use a documented standard that is widely accepted by industry.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do **UML** documents exist that describe the shared interfaces?

Procedure:

Ask for the design documents to be provided during the review process.

Example:

None

2) Test:

Are there **WSDL** files that document the interface to Web services?

Procedure:

Look for the existence of **`.WSDL`** files.

Example:

None

3) Test:

Are there **IDL** files that document the interfaces to **CORBA** services?

Procedure:

Look for the existence of **`.idl`** files.

Example:

None

G1002

Separate public interfaces from implementation.

Rationale:

This guidance encourages clean separation between **interface** and implementation details for all types of application development. This allows components and systems to be **loosely coupled**. The flexibility allows groups of developers to work independently and in parallel to the contract defined by the interface.

Another benefit of hiding implementation details is that it allows the implementation to change without affecting users of the interface. This means the interface can support dynamic and pluggable implementation.

Finally, separating the implementation from the interface allows for version control of the interface separate from the implementation.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Extensibility
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

`C++`: Check to make sure interfaces are defined as pure virtual functions.

Procedure:

Make sure **`C++`** classes are defined in header files. Classes that represent external interfaces should contain only pure virtual functions. Make sure the class does not declare non-constant data members. Also, make sure it does not define default implementation. An interface should provide no default behavior.

Example:

None

2) Test:

`C`: Check to make sure functions are declared in a header file using prototypes.

Procedure:

Make sure each library function has a prototype declaration in the header file.

Example:

None

G1003

Separate shared Application Programming Interfaces (APIs) from internal APIs.

Rationale:

The APIs that are intended to be shared with outside consumers need to remain fairly static in order to facilitate use by the consumers. The consumer and the producer should mutually agree to changes in APIs.

Shared APIs should only have code related to the shared API functionality.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the API contain extraneous interfaces or code that is not required for the API functionality?

Procedure:

Use coverage tool/Junit to make sure there is no extraneous code.

Example:

None.

G1004

Make public interfaces backward-compatible within the constraints of a published deprecation policy.

Rationale:

The public interface is basically a contract between the producer of the functionality defined in an interface and the consumer of the functionality. This and related guidance statements are intended to ensure that this contract remains intact and that the consumer of the functionality is not broken during the update cycle of the interface.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the public interface (interfaces that are used externally, outside the project's domain) contain versioning information?

Procedure:

Check to make sure the interface/class has versioning information.

Example:

None

2) Test:

Does the document structure contain a document that indicates the shelf life of deprecated interfaces?

Procedure:

Check for project documents that have information on the life of deprecated interfaces.

Example:

None

G1008

Isolate platform-specific interfaces.

Rationale:

Insulating platform-specific code (for example code dealing with operating system path conventions) using standard abstractions or custom classes will keep all non-portable code in one place and prevent proliferation of non-portable code throughout the application.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the application contain any platform-specific code that has not been abstracted?

Procedure:

Check code that is non-portable; for instance, the code does not use back slashes (Windows) or forward slashes (UNIX) in literal strings to create a path.

Example:

```
String path = "\tmp";
```

2) Test:

Is platform-specific code isolated into a single class or file?

Procedure:

Search the files for platform-specific code.

Example:

None

G1010

Use open-standard logging frameworks.

Rationale:

Standardizing on one logging **API** means the code will be more portable between developers, and developers no longer need to learn multiple logging frameworks.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

See sublevel guidance: G1209, G1210.

Procedure:

Example:

G1011

Make components independently deployable.

Rationale:

Independently deployable components do not have any dependencies on other components. This is often unattainable because components are often aggregations of lower-level components. Exceptions to this rule can occur if the relationships between components are one or more of the following:

- well-defined and well thought out
- carefully managed
- externally configurable

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Implement a Component-Based Architecture
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Implement a Component-Based Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Implement a Component-Based Architecture
NESI / Part 5: Developer Guidance / Implement a Component-Based Architecture
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is the component dependent on other components?

Procedure:

Check for dependencies.

Example:

None

G1012

Use a set of services to expose Component functionality.

Rationale:

By exposing discrete units of functionality as **services**, business and data integrity remain intact. A service receives a request, processes it, and returns the result to the requester as a single operation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Implement a Component-Based Architecture
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Implement a Component-Based Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Implement a Component-Based Architecture
NESI / Part 5: Developer Guidance / Implement a Component-Based Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are there **WAR** files that contain the component?

Procedure:

Check for the occurrence of **`.war`** files.

Example:

None.

2) Test:

Are there **WSDL** files that define the services?

Procedure:

Check for the occurrence of **`.wsdl`** files.

Example:

None.

G1014

Access databases through open standard interfaces.

Rationale:

The use of non-standard interfaces can cause portability issues. Standards-based database interfaces promote database independence. For example, **Open Database Connectivity** (**ODBC**) is a standard database interface for referencing databases with C/C++ and .NET, while **Java Database Connection** (**JDBC**) is a standard **Application Programming Interface** (**API**) for accessing databases with Java.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are standard interfaces used to access databases?

Procedure:

Check that standards-based interfaces are used to access databases; for example, ODBC for C,C++, or .NET languages, or JDBC for Java.

Example:

None.

G1018

Assign version identifiers to all public interfaces.

Rationale:

Assigning versions is necessary when determining compatibility between the **interface** and its consumer. Versioning public interfaces allows all parties to track the evolution of the interface for backward compatibility. This can help consumers plan for integration and migration. It is important to have the version information in the shared public interface code because it identifies the actual interface to which consumers of the interface will be coding. Another benefit is that it allows tools to generate the documentation automatically so it does not need to be in two places.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the shared public interface code contain versioning information?

Procedure:

Inspect public interfaces or their supporting documentation for version identifiers.

Example:

None.

G1019

Deprecate public interfaces in accordance with a published deprecation policy.

Rationale:

By deprecating instead of removing interfaces, development teams can plan for software migration and continue to run the software with existing (but deprecated) interfaces.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are public interfaces appropriately deprecated?

Procedure:

Check the project documentation for deprecation policy.

Check that interfaces are properly marked and removed according to the deprecation policy.

Example:

None

G1022

Insulate public interfaces from compile-time dependencies.

Rationale:

Compile-time dependencies bind not only the capabilities of the included library, module or object, but also the limitations and vulnerabilities to the software being compiled. If the compiled software is a module that provides a public interface itself, any other software that uses that public interface also assumes the benefits, constraints and risks of the underlying compile-time dependencies. While this can significantly optimize the performance of a module, it can also make use of the public interface difficult if the constraints include hardware architecture limitations or if the vulnerabilities include predictable memory targets for attacks. Later binding techniques (at link time or better yet, run time) can minimize these exposures and maximize flexibility, robustness, interoperability and maintainability.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is the packaging or deployment of the public interface self-contained and isolated to only the public interface(s)?

Procedure:

Check to make sure that the jar, library, assembly, and WSDL only contain the agreed-upon public interface (interfaces being shared externally).

Example:

None

2) Test:

Does the container (jars, libraries, assemblies, WSDL) contain files other than the interface?

Procedure:

Check to make sure the library does not include or rely upon any other files such as resource files, properties files, configuration files, other libraries, XML files, and so on that would force the repackaging of the public interface.

Example:

None

3) Test:

Are there any outside influences that could affect the packaging of the public interface?

Procedure:

Check the public interface for dependence on resource files, properties files, configuration files, XML files, and other libraries or packages.

Example:

None

G1027

Internally document all source code developed with DoD funding.

Rationale:

Well-documented source code is easier to maintain and enhance over time. It is hard enough to get documentation about software and to keep it up to date. If the documentation is not internal to the source code, the chances that the software is current and up-to-date decreases. In recent years, the trend has been to generate external documentation about the software by processing the source code and comments (e.g., **Javadoc**).

In addition to documenting the functionality of the source code, it is important to capture the configuration control information (e.g., CVS).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Standard Interface Documentation
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Standard Interface Documentation
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Standard Interface Documentation
NESI / Part 5: Developer Guidance / Standard Interface Documentation
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do all the source code files have a header that includes a statement protecting government rights to the source code and the right to change the source code?

Procedure:

Scan each file and make sure the header includes a statement that protects the government's right to use, modify, and share the information with other government departments and agencies.

Example:

None

2) Test:

Do all the source code files have a header that includes configuration information?

Procedure:

Scan each file and make sure the header also includes configuration management information such as author, date created, and a history of modifications and versions.

Example:

None

3) Test:

Do all the source code files have internal documentation for attributes, methods that a computer process?

Procedure:

Scan the source files and make sure they are internally documented with tags such as Javadoc or XML tags.

Example:

None

G1030

Use a user interface component library.

Rationale:

User interface component libraries provide a standardized, well-tested look-and-feel without significant development effort. However, care must be taken to ensure that the application code is insulated from dependencies upon a specific UI component library.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Thick Clients
NESI / Part 5: Developer Guidance / User Interfaces / Thick Clients
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the application use a user interface component library?

Procedure:

Check for user interface component library code dependencies in the user interface code.

Example:

None.

G1032

Validate all input fields.

Rationale:

Input validation contributes to data integrity, security, and enhances the end-user experience by detecting errors and preventing problems as close as possible to the point of data entry.
Input validation can be simplified by reducing the number of free form text fields and using selection mechanisms such as radio buttons, option boxes, pull down lists, maps, calendars, clocks, slider bars, and other numeric validation entries.
User input data validation should not be the sole mechanism to ensure data integrity. For example, web applications client -side data validation may be done with javascript, but the user (or an intermediary) may modify or remove the javascript without the knowledge of the server-side web application; therefore it is important to validate input data at both the client-side and server-side.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are all input fields, including non-freeform fields, validated to ensure they can be properly handled across data interfaces: normalized, mediated, and rendered?

Procedure:

Review the code that receives the input fields' data and verify that the inputs are validated against expected interfaces' data models.

Example:

Sample validation techniques:

- validating input data against a white-list of approved values
- validating input data against a black-list of non-allowed values
- validating input data against a regular expression for proper format
- validating input data to not allow inappropriate execution of commands such as used in SQL-Injections attacks

Sample validation tools:

- IBM WebSphere Voice Toolkit VoiceXML validator tool
- Cisco Systems Audium VoiceXML validation for J2EE

G1035

Follow W3C standards for code which will generate a Web page display.

Rationale:

Code cannot be browser-independent if it uses vendor-specific add on features. Vendor-specific add-on features reduce the portability and **interoperability** of the code. Vendor-specific **API**(s) can cause vendor lock-in and in many cases can also cause version lock-in. Following the W3C standards avoids these problems.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the code adhere strictly to the W3C standards?

Procedure:

Check to make sure there is no vendor-specific code.

Example:

None

G1043

Separate formatting from data through the use of style sheets instead of hard coded HTML attributes.

Rationale:

Formatting information will be located in one location instead of scattered throughout each individual Web page of a Web site. This makes a Web site more maintainable.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are any formatting attributes used in any of the HTML tags?

Procedure:

Search all Web pages and make sure there are no formatting attributes such as align, color, font, or size in any tags.

Example:

None

G1044

Comply with Federal accessibility standards contained in Section 508 of the Rehabilitation Act of 1973 (as amended) when developing software user interfaces.

Rationale:

Applicable software must comply with Federal standards to enable better application use for those with disabilities.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Accessibility
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Designing User Interfaces for Accessibility
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Designing User Interfaces for Accessibility
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Accessibility
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do all Web document **HTML**, **JSP**, **ASP**, and **CSS** follow the Disability Act guidelines?

Procedure:

Check to make sure all Web documents follow the guidelines.

Use available validation tools to validate Section 508 accessibility and WAI accessibility. Go to http://www.contentquality.com/Default.asp to validate the page.

Example:

None

G1045

Define XML format information separately in XSL.

Rationale:

XML documents should be free of any presentation information and should only contain data. Separating presentation data from content allows multiple presentations for the same content data.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / XML Rendering
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / XML Rendering
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Check for presentation information in XML documents?

Procedure:

Does the XML document contain only data?

If the XML document is not an document, does it contain presentation information?

Example:

None

G1050

In ASP, isolate the presentation tier from the middle tier using COM objects.

Rationale:

This is the best way to isolate the presentation tier from the middle tier in ASP.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Active Server Pages (ASP)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Active Server Pages (ASP)
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Is all the middle tier code isolated from the presentation tier in ASP via COM?

Procedure:

Verify that ASP files do not contain middle-tier code. Instead, this code should be in COM objects referenced from the ASP.

Example:

None

G1052

Use the code-behind feature in ASP.NET to separate presentation code from the business logic.

Rationale:

Separating presentation code from business logic allows the developers and content designers to work independently. It also makes the code more maintainable because changes in the design elements or business elements do not affect each other.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Is there code in ASP pages?

Procedure:

Check to make sure that ASP files have the code-behind attribute in the first line instead of embedded C# code in the ASP.

Example:

None

G1053

Do not embed HTML code in any code-behind code used by aspx pages.

Rationale:

Intermixing VB or C# or C++ with presentation code (HTML) makes the code unnecessarily difficult to maintain by both the developer and designer. This is similar in concept to Java's not embedding HTML code in **servlets**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Check for HTML code in code-behind code.

Procedure:

Check the code-behind file (**`.aspx.vb`** for example) for any HTML tags.

Example:

None

G1056

Specify a versioning policy for .NET assemblies.

Rationale:

Versioning assemblies and configuring dependent assemblies allow the **Common Language Runtime** (**CLR**) to load the proper assemblies at runtime for an application. This insulates the application from configuration changes.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the application assembly have versioning information?

Procedure:

Check the application assembly manifest for versioning information.

Use the .NET configuration tool to check for versioning policy and versioning information.

Example:

None

G1058

Use the Model, View, Controller (MVC) pattern to decouple presentation code from other tiers.

Rationale:

Separating data-layer code from presentation-layer code provides the ability to base multiple views on the same model. This is especially important in the enterprise model because often, the user interface varies with the device (browser, mobile phone, thick client, etc.).

Isolating different layers allows changes to occur in each layer without impacting other layers. For instance, if the data layer (model) decides to switch databases, the changes are isolated to the data layer and do not affect the view layer or controller layer.

Lastly, because MVC architecture enforces separation between presentation, processing, and data layer, this allows functionality to be loosely coupled and therefore more suited for reuse.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Active Server Pages (ASP)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Active Server Pages (ASP)
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Active Server Pages for .NET (ASP.NET)
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Java Server Pages (JSP)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Java Server Pages (JSP)
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the application enforce clear separation between data layer (model), presentation layer (view), and middle/business layer (controller)?

Procedure:

Ensure that all page renderings use a Model-View-Controller (MVC) pattern using, for example, **JavaServer Pages** (**JSPs**) and **servlets** or ASP.NET pages and Code Behind files.

Example:

None.

G1060

Encapsulate Java code that is used in JSP(s) in tag libraries.

Rationale:

Separating code from presentation allows developers and designers to work independently. It makes the code reusable and more maintainable because it is defined in a tag library.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Java Server Pages (JSP)
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Java Server Pages (JSP)
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do the JSP pages use tag libraries?

Procedure:

Look through the JSP pages for embedded Java source code.

Example:

None

G1071

Use vendor-neutral interface connections to the enterprise (e.g., LDAP, JNDI, JMS, databases).

Rationale:

Increase **portability** and maintainability. Many of the newer connection mechanisms are vendor-neutral. Use these instead of isolation design patterns or vendor-specific connection mechanisms.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is the connection mechanism vendor-neutral?

Procedure:

Examine the source code for vendor-specific imports or includes. Use only standard APIs.

Example:

None

G1073

Isolate vendor extensions to enterprise-services standard interfaces.

Rationale:

Vendor extensions are convenient but help create "vendor lock" and reduce vendor neutrality and migration. It is best to avoid these extensions altogether. If that is not possible, then isolate them in an **adapter** or a wrapper-like construct.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are vendor extensions to enterprise services used?

Procedure:

Make sure that no vendor-specific code is included or imported except as part of an adapter or wrapper.

Example:

None

G1078

Document the use of non-Java EE-defined deployment descriptors.

Rationale:

Deployment descriptors that are not defined by the J2EE specification are not portable between **application servers**. For example, BEA WebLogic has a vendor-specific deployment descriptor called **`weblogic-ejb-jar.xml`** and JBoss has a vendor specific deployment descriptor called **`jboss-jar.xml`** .

Referenced By:

NESI / Part 5: Developer Guidance / Middleware / Java EE Deployment Descriptors
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are all the XML files that are not part of the Java EE specification identified in a delivered document?

Procedure:

Search all XML documents in the META-INF and WEB-INF directories and identify any XML files that are not defined by Java EE. These files should be in a README or other delivered file that describes their purpose:

Example:

Web application	**`WEB-INF/web.xml`**
EJB JAR	**`META-INF/ejb-jar.xml`**
J2EE Connector	**`META-INF/ra.xml`**
Client application	**`META-INF/application-client.xml`**
Enterprise application	**`META-INF/application.xml`**

G1079

Isolate tailorable data values into the deployment descriptors for Java EE applications.

Rationale:

Do not hard-code tailorable data into source files. The standard location for tailorable data for Java EE applications is in deployment descriptors. Developers should not "reinvent the wheel" by creating a non-standard mechanism for retrieving configurable data. Make tailorable data accessible through application contexts provided by the application **container** (**Java EE application server**).

Referenced By:

NESI / Part 5: Developer Guidance / Middleware / Java EE Deployment Descriptors
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Is tailorable data configured using deployment descriptors?

Procedure:

Check the deployment descriptor for instances of tailorable data.

Example:

Name-value pairs such as **environment variables** configured using resource-env-ref elements.

JNDI locations configured using resource-ref elements.

G1080

Adhere to the Web Services Interoperability Organization (WS-I) Basic Profile specification for Web service environments.

Rationale:

Most of the **COTS** Web service products have already met this requirement. This is intended to cause a rejection of the non-standard Web server.

The WS-I Basic Profile specification is available from the Web Services Interoperability Organization Web site: WS-I Org Basic Profile.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Web Services Compliance
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Web Services Compliance
NESI / Part 5: Developer Guidance / Middleware / Web Services / Web Services Compliance
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is the Web service product WS-I Basic Profile specification compliant?

Procedure:

Identify the Web service product being used, and verify through a literature search that it is WS-I Basic Profile specification compliant.

Example:

None

G1082

Use the document-literal style for all data transferred using SOAP where the document uses the World Wide Web Consortium (W3C) Document Object Model (DOM).

Rationale:

The document-literal style requires defining the input and output parameters to a Web service as documents that follow the W3C Document Object Model (DOM). The DOM acts as a contract between the producer and the consumer of the Web service that is formal, well-defined, and rigorous. Validating the DOM against an **XML** Schema Definition (**XSD**) can help resolve discrepancies in the interface.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Web Services Compliance
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Web Services Compliance
NESI / Part 5: Developer Guidance / Middleware / Web Services / Web Services Compliance
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / SOAP
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / SOAP
NESI / Part 5: Developer Guidance / Middleware / Web Services / SOAP
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the **WSDL** define input, output, or returned parameters as Documents that follow the **W3C** Document Object Model (**DOM**)?

Procedure:

Review all WSDL files used to describe a Web service, and make sure they only pass documents. Document types should be **`xsd:anyType`**.

Example:

None

G1083

Do not pass Web Services-Interoperability Organization (WS-I) Document Object Model (DOM) documents as strings.

Rationale:

Because of the relative simplicity of converting an **XML** document to a string, it is easy to pass an entire document as a string rather than as an XML document. This can cause problems if the document contains tags that are similar to the tags used in the **SOAP**. Passing it as an XML document ensures that the document is treated as a single entity.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Web Services Compliance
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Web Services Compliance
NESI / Part 5: Developer Guidance / Middleware / Web Services / Web Services Compliance
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the **WSDL** define input, output, or returned parameters as strings?

Procedure:

Review all the WSDL files used to describe a Web service and make sure that they only pass documents, not strings. Document types should be **`xsd:anyType`**.

Example:

None

G1085

Establish a registered namespace in the XML Gallery in the DoD Metadata Registry for all DoD Programs.

Rationale:

A **registered namespace** permits unique identification and categorization of a Program which avoids name collisions and conflicts. The DoD Net-Centric Data Strategy requires storing data products in shared spaces to provide access to all authorized users and tagging these data products with **metadata** to enable discovery of data by authorized users. The use of a unique **registered namespace** provides an absolute identifier to products associated with a particular product and is an **XSD** schema requirement.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / WSDL
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / WSDL
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / WSDL
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / WSDL
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / WSDL
NESI / Part 5: Developer Guidance / Middleware / Web Services / WSDL
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Program have an assigned namespace in the **DoD Metadata Registry**?

Procedure:

Check the **DoD Metadata Registry** to determine whether program is associated with **COI**(s).

Example:

None

G1087

Validate all Web Services Definition Language (WSDL) files that describe Web services.

Rationale:

Manually editing a **WSDL** file is error-prone, work-intensive, and hard to maintain. However, if the user wants to do it, there is no way to detect a manually edited file from one that was auto generated. The important thing is not how the **WSDL** file is generated but rather that the **WSDL** file is valid. It must be validated with a **WSDL** validator.

Note: Not all **WSDL** files that are generated and valid are necessarily interoperable.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / WSDL
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / WSDL
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / WSDL
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / WSDL
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / WSDL
NESI / Part 5: Developer Guidance / Middleware / Web Services / WSDL
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Insulation and Structure
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Insulation and Structure
NESI / Part 5: Developer Guidance / Middleware / Web Services / Insulation and Structure
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services
NESI / Part 5: Developer Guidance / Middleware / Web Services

Evaluation Criteria:

1) Test:

Can the **WSDL** file be validated?

Procedure:

Download a validation tool and test WSDL files.

Example:

Sample tools:

WS-I Organization:	http://www.ws-i.org/deliverables/workinggroup.aspx?wg=testingtools
Eclipse:	http://dev.eclipse.org/viewcvs/indextech.cgi/wsvt-home/main.html?rev=1.20
XMethods:	http://xmethods.net/ve2/Tools.po

Pocket Soap:	http://pocketsoap.com/wsdl/

G1088

Use isolation design patterns to define system functionality that manipulates Web services.

Rationale:

Insulating **SOAP** Web-service manipulation using standard abstraction patterns such as a **proxy** or **adapter** insulates the software system from changes in the Web service interface and promotes maintainability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Insulation and Structure
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Insulation and Structure
NESI / Part 5: Developer Guidance / Middleware / Web Services / Insulation and Structure
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / SOAP
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / SOAP
NESI / Part 5: Developer Guidance / Middleware / Web Services / SOAP
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services
NESI / Part 5: Developer Guidance / Middleware / Web Services

Evaluation Criteria:

1) Test:

Are Web service calls isolated in a single adapter or proxy object?

Procedure:

Check to see if all Web service calls are isolated to a single adapter or proxy object.

Example:

None

2) Test:

Are Web service calls inside of the application code?

Procedure:

Check for proliferation of Web service calls inside an application.

Example:

None

3) Test:

Are SOAP-client calls inside the application code?

Procedure:

Check to see if SOAP-client code is proliferated inside the application code?

Example:

None

G1090

Do not hard-code a Web service's endpoint.

Rationale:

An endpoint is the URL or location of the **Web service** on the **Internet**. A major benefit of Web services is the ability to relocate a Web service to another location or dynamically discover and use a Web service using registry facilities. Some Web service vendors hard-code the URL of the Web service which causes maintenance and portability problems.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Insulation and Structure
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Insulation and Structure
NESI / Part 5: Developer Guidance / Middleware / Web Services / Insulation and Structure
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services
NESI / Part 5: Developer Guidance / Middleware / Web Services

Evaluation Criteria:

1) Test:

Are there any hard-coded URLs in the client-side code?

Procedure:

Parse the client code looking for hard-coded URLs.

Example:

The Java code samples below illustrate how this might be done. The first sample shows parameters that are hard-coded; the second sample shows how parameters and Web service endpoints are insulated.

1. Hard-coded parameters:

```
// Sample code that has hard-coded parameters
// before applying insulation
public static void main
  ( String[] args
  ) throws Exception
{ //The SOAP endpoint
  String sSoapEndpoint
     = "http://live.capescience.com:80"
       + "/ccx/AirportWeather";
   AirportWeatherClient myProxy = null;
  try
  { myProxy
     = AirportWeatherClientFactory.create
        ( sSoapEndpoint);
   System.out.println
     ("Location: "
      + myProxy.getLocation(args[0])
     );
```

```
   //rest of code removed for brevity
  } // End try
  Catch ( Exception exception )
  { System.out.println("Error: " + exception);
  } // End catch
};//end of main program
```

2. Insulated parameters and Web service endpoints

a. Property file - this code shows the property file itself:

c. Client sample code:

```
import java.io.*;
import java.rmi.*;
import java.util.*;
import AirportWeatherClient; // auto-generated SOAP
                             // client from IDE */
public class WeatherProxy
  implements airportWeatherProxy
{
 //
 //code removed for brevity
 //
 public WeatherProxy
  ( String propFileStr )
 { try
   { getEndPoint(propFileStr);
   } // End try
   catch(Exception e)
   { // Handle exception here
   } // End catch
   connect2SOAP();
 }// End constructor
 /* public api's */
 public String getLocation()
 { return location;
 } // End getLocation
   . . . // Other public API's removed for brevity
 private void getEndPoint
  ( String propsFile )
  throws Exception
 { if ( propsFile == null || propsFile.length() == 0 )
   { throw new Exception
       ( "SOAP EndPoint parameter not defined");
   } // End if
   props = new Properties();
   try
   { InputStream is = new FileInputStream(propsFile);
     props.load(is);
     is.close();
    } // End try
    catch ( Exception exception )
    { throw new Exception
        ( "can't read props file " + propsFile);
    } // End catch
    Enumeration enum = props.propertyNames();
    while ( enum.hasMoreElements() )
    { String endPointString = null;
      String propName = enum.nextElement().toString();
      if ( propName.equals ( endPointString ) )
     { soapEndpoint = props.getProperty( propName );
       break;
     } // end if
    } // End while
  }//end getEndPoint
 private void connect2SOAP()
 { try
   { myProxy
       = AirportWeatherClientFactory.create
           ( soapEndpoint );
     . . . //code removed for brevity
   } // End try
```

```
   catch ( Exception exception )
   { System.out.println
       ( "Error connecting to SOAP server: "
          + exception
       );
   } // End catch
 } // End connect2SOAP
  private Properties props = null;
  private String propsFile = null;
  private AirportWeatherClient myProxy = null;
  private String soapEndpoint = null;
  private String location = null;
}//end WeatherProxy
public class Weather
{ private static WeatherProxy myWeatherProxy = null;
  public static void main
    ( String[] args
    ) throws Exception
  { try
    { myWeatherProxy = new WeatherProxy ( args[0] );
    } // End try
    Catch ( Exception exception )
    { throw new Exception
        ( "can't connect to SOAP server");
    } // End catch
    System.out.println
      ( "Location: "
        + myWeatherProxy.getLocation()
      );
    . . . //code deleted for brevity
  }//end main
}//end Weather
```

G1093

Implement exception handlers for SOAP-based Web services.

Rationale:

SOAP exceptions result when there are connectivity problems or violations in the SOAP protocol between the client and the server.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / SOAP
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / SOAP
NESI / Part 5: Developer Guidance / Middleware / Web Services / SOAP
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the Web application client have exception handlers for **`SOAPExceptions`**.

Procedure:

Check to see that the Web application client has an exception block specifically for **`SOAPException`**.

Example:

None

2) Test:

Does the Web application client test the SOAP response for a fault?

Procedure:

Verify the Web application client handles a true value returned from the **`response.generatedFault`**.

Example:

None

G1094

Catch all exceptions for application code exposed as a Web service.

Rationale:

Any exception can reveal system internals and thus compromise security. Also, internal exceptions are not user friendly.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Does each exposed Web method catch all possible exceptions and re-throw a declared application exception?

Procedure:

Verify that each exposed Web method has an exception block that catches all possible exceptions and then re-throws them as a declared application exceptions.

Example:

None

2) Test:

Does each exposed Web method catch all possible runtime exceptions and re-throw a declared application runtime exception?

Procedure:

Verify that each exposed Web method has an exception block that catches all possible exceptions and then re-throws them as a declared application exceptions.

Example:

None

G1095

Use W3C fault codes for all SOAP faults.

Rationale:

Having predefined and accepted fault codes allows consumers to handle SOAP faults appropriately without prior knowledge of custom fault codes.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / SOAP
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / SOAP
NESI / Part 5: Developer Guidance / Middleware / Web Services / SOAP
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the Web application throw fault codes from the accepted list of fault codes?

Procedure:

Verify that each fault code thrown by the Web application is from the accepted list of SOAP fault codes defined by the W3C.

Example:

None

G1118

Localize CORBA vendor-specific source code into separate modules.

Rationale:

The general guidance is to minimize CORBA vendor-specific source code, while recognizing that vendor-specific features are necessary in certain circumstances. However, isolating vendor-specific code reduces maintenance effort.

Vendor capabilities tend to change more rapidly than CORBA-standard specifications. Experience shows that vendor updates frequently require modification to application source code, due to changing vendor interface conventions. These modifications impose vendor-version-specific constraints on the application, thereby complicating maintenance.

Example

Encapsulating CORBA ORB operations

The following examples show how to encapsulate binding operations for a C++ ORB, and naming service operations for a Java ORB.

C++ ORB binder template

The code below shows a sample template for binding to the C++ ORB. IONA's ORBIX was used in this example.

```
/* ====================================================
ServerBinder.h (Template)
this is a generic binder to ORBIX
==================================================== */
#ifndef _BINDER_H_
#define _BINDER_H_
#ifndef IOSTREAM_H
#define IOSTREAM_H
#include <iostream.h>
#endif
#ifndef STDLIB_H
#define STDLIB_H
#include <stdlib.h>
#endif
template <class SERVERNAME, class VARPTR>
class Binder
{ private:
    char* serverName;
  public:
    Binder(char* svName):serverName(svName){};
    ~Binder(){};
    int bind( VARPTR* p)
    { int attempts = 0, success = 0;
      int maxtries = 5, retval = 0;
      while ( ( attempts < maxtries )
             && (!success)
            )
      { ++attempts;
        cout << "Binding to server, attempt "
            << attempts
            << endl;
        try
        { (*p) = SERVERNAME::_bind();
          cout << "Bound to server"
              << endl;
          success = retval = 1;
        } // End try
        catch ( CORBA::SystemException &systemException )
        { cout << "SystemException, ServerBinder::bind"
              << endl
              << systemException;
```

```
          success = 1;
          retval = 0;
        } // End catch SystemException
         catch (...)
         { cout << "unknown Exception, ServerBinder::bind"
               << endl;
           success = 1;
           retval = 0;
         } // End catch all
      } //end while
      return retval;
    } //end bind
} //end Binder
#endif
```

Ada ORB binder template for C++

The code below shows a C++ template for binding to an Ada ORB. ORBexpress was used in this example.

```
/* ====================================================
ada_binder.h (Template)
this is a generic binder to ORBExpress
==================================================== */
#ifndef _ADA_BINDER_H_
#define _ADA_BINDER_H_
#ifndef IOSTREAM_H
#define IOSTREAM_H
#include <iostream.h>
#endif
#ifndef STDLIB_H
#define STDLIB_H
#include <stdlib.h>
#endif
template <class SERVERNAME, class VARPTR >
class Ada_Binder
{ private:
    char* adaIorString;
  public:
     Ada_Binder
       ( char* iorString)
       : adaIorString ( iorString )
     {};
     ~Ada_Binder(){};
     int bindToAda( VARPTR* p)
     { int attempts = 0, success = 0;
       int maxtries = 5, retval = 0;
       while ( ( attempts < maxtries)
              && (!success)
             )
       { ++attempts;
         cout << "Binding to server, attempt "
             << attempts
             << endl;
         try
         { cout <<"adaIorString:"
               << endl
               << adaIorString
               << endl;
               (*p) = SERVERNAME::_bind(adaIorString);
//can't use string_to_object in this version
//it kills the ada IOR
//            CORBA::Object_ptr myptr
              CORBA::Orbix.string_to_object
                ( adaIorString );
//            (*p) = SERVERNAME::_narrow(myptr);
            cout << "Bound to server" << endl;
            success = retval = 1;
         } // End try
         catch (CORBA::SystemException& systemException)
         { cout << "SystemException, "
               << "AdaServerBinder::bind"
               << endl
```

```
                << systemException;
            success = 1;
            retval = 0;
          } // End SystemException
          catch (...)
          { cout << "Unknown Exception, "
                << "AdaServerBinder::bind"
                << endl;
            success = 1;
            retval = 0;
          } // End catch all
       } // end while
       return retval;
     } // end bind
} // end ADA_Binder
#endif
```

Example

Naming service operations for a Java ORB

Java helper class

This example is a helper class, JavaNamingHelper.java, that encapsulates CORBA naming service operations for all services to use. We used Java JDK 1.4 ORB to create this example.

```
import java.util.*;
import org.omg.CORBA.*;
import org.omg.CORBA.ORB.*;
import org.omg.CORBA_2_3.ORB.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContext.*;
import org.omg.CosNaming.NamingContextPackage.*;
import CBRNSensors.JSLSCAD.*;
public class JavaNamingHelper
{ static NamingContext nameSvc = null;
  static org.omg.CORBA.Object objref = null;
  static JSLSCADSensor myCBRNSensor = null;
  static org.omg.CORBA.Object myobj = null;
  public JavaNamingHelper()
  {
  }
  private static void showNamingContext
    ( org.omg.CORBA.ORB myorb )
  {
  public static NamingContext getNamingSvc
    ( org.omg.CORBA.ORB lclorb,
      String nameSvcName
    )
  { NamingContext lclNameSvc = null;
    try
    { org.omg.CORBA.Object nameSvcObj
        = lclorb.resolve_initial_references
            ( "NameService" );
       // . . . other business logic removed
       //       for brevity
    } // End try
    catch(org.omg.CORBA.COMM_FAILURE cf)
    { . . . // error code goes here
    } // End cstch
    catch ( org.omg.CORBA.ORBPackage.InvalidName invalidName)
    { . . . // error code goes here
    } // End catch
    catch ( SystemException systemException )
    { . . .// error code goes here
    }
  } // End getNamingSvc
  public static org.omg.CORBA.Object getObjFromNameSvc
    ( org.omg.CORBA.ORB myorb,
      String targetSensorName
    )
  { . . . // business logic goes here
```

```
  } //end getObjFromNameSvc
  public static int setObj2NameSvc
    ( org.omg.CORBA.ORB myorb,
      BasesSensor mySensor,
      String targetSensorName
    )
  {. . . // business logic goes here
  }//end setObj2NameSvc
}; //end class JavaNamingHelper
```

Java server implementation

The code below is a sample Java server implementation that uses the naming service helper class.

```
import java.io.*;
import java.util.*;
import org.omg.CORBA.*;
import org.omg.CORBA.ORB.*;
import org.omg.CORBA_2_3.ORB.*;
import org.omg.PortableServer.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContext.*;
import org.omg.CosNaming.NamingContextPackage.*;
class MyServer
{ public static Properties props;
  public static ORB myorb = null;
  public static NamingContext nameSvc = null;
  public static RootSensor mySensor = null;
  public static String propertyFilePath = null;
  public static final String MY_SENSOR_NAME = "MYSENSOR";
  static public void main(String[] args)
  { // handle arguments
    System.out.println(" CORBA Server starting...\n");
    try
    { // Initialize the ORB.
      myorb = ORB.init(args, props);
      //instantiate servant and create ref
      POA rootPOA
        = POAHelper.narrow(myorb.resolve_initial_references
           ( "RootPOA" );
     . . . // rest of initialization code goes here
    } // End try
    catch ( org.omg.CORBA.ORBPackage.InvalidName invalidName )
    { . . . //error code goes here
    } // End invalidName
    // other exception types to catch go here
    catch ( SystemException systemException)
    {  System.err.println ( systemException );
    } // End systemException
    // naming service hookup
    JavaNamingHelper.setObj2NameSvc
      ( myorb,mySensor,
        MY_SENSOR_NAME
      );
    try
    { System.out.println(" Ready to service requests\n");
      myorb.run();
    } // End try
    catch(SystemException systemException)
    { System.err.println ( systemException );
    } // End catch systemException
  } // End static block
} // End MyServer
```

Java client implementation

The code below is a sample client implementation that uses the naming service helper class.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are any non-CORBA compliant CORBA:: objects declared or defined in the module?

Procedure:

Review the code for a service that can be used to obtain configuration.

Example:

None

2) Test:

Does the module contain vendor names anywhere in code text?

Procedure:

Review the code looking for a service that can be used to obtain configuration.

Example:

None

G1119

Isolate user-modifiable configuration parameters from the CORBA application source code.

Rationale:

Configuration parameters control the behavior of the CORBA **ORB** service environment and client/service processes during startup, execution, and termination. This parameterization allows execution-time control modification without having to rebuild, reinstall, or redeploy.

Configuration defines the state of the client-and-service environment throughout the lifetime of the processes involved. This relates to considerations such as the allocation of threading and resources, **POA** policies, the instantiation of servants and their invocations, failure and security behavior, connection management, quality of service prioritization, and so forth. The point is that CORBA provides an extremely complex but flexible environment for distributed computing interaction. Consequently, the designer requires flexible guidance to handle this option-rich environment.

Configuration processes and their related parameters fall into two categories. The first involves configuration matters, which are defined to be perpetually static by the system architecture. The second involves matters that are intended to be modifiable by users.

The first category, immutable configuration settings, relates to fundamental underlying assumptions that are foundational for the implementation. These are matters for which no user modification is ever intended as it would lead to unspecified behavior. Consider the example of a service implementation that is programmed to be single threaded. In this case, multi-threading controls are irrelevant and multiple instantiation would lead to dangerous confusion. For immutable configuration parameters, localized and well-commented implementation in the application source code is appropriate.

For user-modifiable configuration settings, there are two further by-design divisions. The first involves configuration settings that are intended to be accessible by distributed processes. The second involves host-specific settings which relate to resources locally available, for which remote access is not desired. These are discussed in the related sublevel guidance

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are configuration parameters isolated from CORBA application code?

Procedure:

Check source code for configurable parameters to verify that such parameters are not hard-coded within the code and are configurable within configuration files.

Example:

G1121

Do not modify CORBA Interface Definition Language (IDL) compiler auto-generated stubs and skeletons.

Rationale:

The purpose of the IDL auto-generated stub and skeleton files is to provide a source code facility/mechanism for the developer in a specific language to use the IDL-described object interface in that specific language. The internal content of these files changes with the application's IDL modification, with IDL compiler-environment configuration settings, and with vendor-product compiler and **ORB** upgrades. By design, these files are not intended to be modified by the application developer. Developer modification of any auto-generated stub or skeleton file will typically lead to very severe maintenance hazards and failed application rebuild results.

The stub files describe the language source-code interface from the client side. Their use involves including the client stub header in the application's call invocation code.

The skeleton files describe the language source code interface from the service implementation side. Their use involves including the skeleton header in the application's operator implementation code. Their use also requires developer modification of a renamed clone of the auto-generated skeleton body file. These techniques are described in every ORB vendor's programming reference manuals.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Is any application code contained in the auto-generated code?

Procedure:

Inspect the auto-generated file creation/modification dates to verify that no tampering occurred after the IDL compilation step in the build process.

Example:

The following examples are all based upon a single CORBA IDL interface.

MyIdlInterface.idl

```
interface MyIdlInterface
{
  readonly attribute string version;
  void stop();
  void start();
  string error();
}; // End MyIdlInterface
```

ORBExpress compiler

The ORBExpress IDL compiler generates these files:

- **`myIdlInterface.h`** - Client-side stub header
- **`myIdlInterface.cxx`** - Client-side stub implementation

- **`MyIdlInterface_s.h`** - Abstract servant header
- **`MyIdlInterface_s.cxx`** - Abstract servant implementation
- **`MyIdlInterface_impl.h`** - Server implementation header
- **`MyIdlInterface_impl.cxx`** - Server implementation implementation

> ***Note:*** *The only files that should be edited are* ***`MyIdlInterface_impl.h`*** *and* ***`MyIdlInterface_impl.cxx`*** *. The IDL compiler checks for the existence of the implementation (i.e. _impl) files and will not overwrite them.*

MyIdlInterface_impl.cxx

```
// Generated for interface MyIdlInterface
// in myIdlInterface.idl
#include "MyIdlInterface_impl.h"
MyIdlInterface_impl::MyIdlInterface_impl
  ( PortableServer::POA* oe_poa,
    const char* oe_object_id
  ) : POA_MyIdlInterface
        ( oe_object_id,
          oe_poa
        )
{ . . . // TO DO: add implementation code here
} // emd constructor
MyIdlInterface_impl::MyIdlInterface_impl
  ( const MyIdlInterface_impl& obj )
  : POA_MyIdlInterface(obj)
{ . . . // TO DO: add implementation code here
} // End constructor
MyIdlInterface_impl::~MyIdlInterface_impl()
{ . . . // TO DO: add implementation code here
} // End destructor
CORBA::Char* MyIdlInterface_impl::version
  ( CORBA::Environment& _env )
{ return CORBA::string_dup(_version);
} // End version
void MyIdlInterface_impl::stop
   ( CORBA::Environment& _env )
{ . . . // TO DO: add implementation code here
} // End stop
void MyIdlInterface_impl::start
   ( CORBA::Environment& _env )
{ . . . // TO DO: add implementation code here
} // End start
CORBA::Char* MyIdlInterface_impl::error
  ( CORBA::Environment& _env )
{ CORBA::Char* result;
  . . . // TO DO: add implementation code here
  return result;
} // End error
```

Java JDK compiler

The Java JDK IDL compiler generates these files:

- **`MyIdlInterface.java`**
- **`MyIdlInterfaceHelper.java`**
- **`MyIdlInterfaceHolder.java`**
- **`MyIdlInterfaceOperations.java`**
- **`MyIdlInterfacePOA.java`**
- **`_MyIdlInterfaceStub.java`**

MyIdlInterfacePOA.java

```
/**
 * MyIdlInterfacePOA.java .
 * Generated by the IDL-to-Java compiler
```

```
 * (portable), version "3.1"
 * from myIdlInterface.idl
 */
public abstract class MyIdlInterfacePOA
  extends org.omg.PortableServer.Servant
  implements MyIdlInterfaceOperations,
             org.omg.CORBA.portable.InvokeHandler
{ . . . // rest of the auto-generated code removed for brevity
} // End MyIdlInterfacePOA
```

MyIdlInterfaceImpl.java

```
package myIdlImpl;
import org.omg.CORBA.*;
import org.omg.CORBA.ORB.*;
import org.omg.CORBA_2_3.ORB.*;
import org.omg.PortableServer.*;
public class  MyIdlInterfaceImpl
  extends  MyIdlInterfacePOA
{
  private String strVersion;
  private String errString;
  public String version ()
  { . . . // implementation code goes here
    return strVersion;
  } // End version
  public void stop ()
  { . . . // implementation code goes here
  } // End stop
  public void start ()
  { . . . // implementation code goes here
  } // End start
  public String error ()
  {. . . // implementation code goes here
   return errString;
  } // End error
} // End MyIdlInterfaceImpl
```

G1123

Use the Fat Operation Technique in IDL operator invocation.

Rationale:

This reduces the CORBA messaging overhead. The performance cost of network CORBA messaging is determined by two factors: latency and marshaling rate. Call latency is the minimum cost of sending any message at all. The marshaling rate is determined by the sizes of sending and receiving parameters and of return values.

In the situation of a large number of objects involving objects that hold a small amount of stat, the call latency cost far exceeds the marshalling costs. Taking advantage of this reality, the "Fat Operation Technique" involves constructing structure objects which hold an aggregation of related attributes, and using the resulting structures in operation invocation parameters and returns. This amounts to transferring a larger amount of information with each network transaction.

For more information, see "Advanced CORBA Programming with C++" by Henning & Vinoski, 1999 Addison Wesley, Chapter 22.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the IDL contain function calls which have structure objects that are passed as parameters or returned from operators?

Procedure:

Inspect the IDL file and manually check for parameters or returns using objects defined as structures, and verify that they are passed from methods also declared in the IDL.

Example:

None

G1125

Use the Department of Defense Metadata Specification (DDMS) for standardized tags and taxonomies.

Rationale:

These standardized tags or Metacards will be developed, maintained, and placed under configuration as appropriate and will comply with the **DDMS** and **COI** guidance. These include specifications defining the tagging for security classification and dissemination control. See the DoD Discovery Metadata Specification Web site (http://metadata.dod.mil/mdr/irs/DDMS/) for the current **DDMS** standards.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Has the Program documented the profile used for published data assets in accordance with guidance?

Procedure:

Check the DoD Metadata Registry to determine whether the program is associated with **COI**(s).

Example:

None

G1127

Use a UDDI specification that supports publishing discovery services.

Rationale:

UDDI provides a registration for services, and the **OASIS** UDDI 2.0 specification has become a standard method for publishing discovery services.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 5: Developer Guidance / Middleware / Web Services / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are the Web services registered in a **UDDI** registry?

Procedure:

Verify the registration in the UDDI registry.

Example:

None

2) Test:

Is the registry **UDDI** 2.0 or higher?

Procedure:

Determine if the particular UDDI registry is UDDI Version 2.0 or higher.

Example:

None

G1131

Use industry standard Universal Description, Discovery, and Integration (UDDI) APIs for all UDDI inquiries.

Rationale:

There is a standard **API** that uses **SOAP** messages to communicate with the UDDI registry. To increase compatibility and portability, use this API exclusively.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 5: Developer Guidance / Middleware / Web Services / Universal Description, Discovery, and Integration (UDDI)
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are all the interfaces to the UDDI registry made using the UDDI standard API?

Procedure:

The standard API for UDDI is SOAP based. Requests and responses are passed using documents. Test the traffic flow between the client and the UDDI registry for messages that are defined in the UDDI specification. Use standard libraries to send and receive the messages (e.g., JUDDI for Java).

Checking for the use of packages like JUDDI does not require the application to be running.

Example:

The following is an example as provided in the UDDI API reference: http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.htm#_Toc25137712 .

find_binding

The find_binding API call returns a **`bindingDetail`** message that contains zero or more binding Template structures matching the criteria specified in the argument list.
Syntax

Syntax

Arguments

serviceKey	This uuid_key is used to specify a particular instance of a businessService element in the registered data. Only bindings in the specific businessService data identified by the serviceKey passed will be searched.
maxRows	This optional integer value allows the requesting program to limit the number of results returned.
findQualifiers	This optional collection of findQualifier elements can be used to alter the default behavior of search functionality. See the findQualifiers appendix for more information.
tModelBag	This is a list of tModel uuid_key values that represents the technical fingerprint of a bindingTemplate structure contained within the businessService specified by the serviceKey value. Only bindingTemplates that contain all of the tModel keys specified will be returned (logical AND). The order of the keys in the tModel bag is not relevant.

Returns

This API call returns a **`bindingDetail`** message upon success. In the event that no matches were located for the specified criteria, the **`bindingDetail`** structure returned will be empty (i.e., it contains no bindingTemplate data.) This signifies a zero match result. If no arguments are passed, a zero-match result set will be returned.
In the event of an overly large number of matches (as determined by each Operator Site), or if the number of matches exceeds the value of the **`maxRows`** attribute, the Operator site will truncate the result set. If this occurs, the response message will contain the truncated attribute with the value "true".

Caveats

If any error occurs in processing this API call, a **`dispositionReport`** element will be returned to the caller within a SOAP Fault. The following error number information will be relevant:

E_invalidKeyPassed	This signifies that the uuid_key value passed did not match with any known serviceKey or tModelKey values. The error structure will signify which condition occurred first, and the invalid key will be indicated clearly in text.
E_unsupported	This signifies that one of the findQualifier values passed was invalid. The invalid qualifier will be indicated clearly in text.

G1132

Implement the data tier using commercial off-the-shelf (COTS) relational database management system (RDBMS) products that implement the SQL standard.

Rationale:

COTS RDBMS products are technically mature, and their capabilities are continually expanding (to include capabilities such as row-level locking, stored procedures, triggers, and high-level language interfaces). Moreover, there is a large technical community able to develop and maintain data systems based on these products. It is likely that a COTS RDBMS will provide many of the data tier capabilities a developer requires.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is the proposed COTS RDBMS product a readily available and supportable COTS product that implements the SQL standard?

Procedure:

Verify that the COTS RDBMS product is widely in use in the DoD environment (e.g., Oracle, SQL Server, or DB2), has a large support community, and is likely to be supported for the lifecycle of the project.

Example:

None

G1141

Base new data models on existing data models developed by Communities of Interest (COI).

Rationale:

Using COI-developed **data models**, or portions thereof, supports interoperability among systems through the use of common semantics.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Are new data models based on COI-developed data models?

Procedure:

Determine whether a COI exists for the technical areas accommodated in the system requirements. Verify that new data models are based on data models the relevant COIs have developed.

Example:

The Universal Core (UCore) data model, Joint Command Control and Consultation Exchange Data Mode (JC3IEDM), and the National Information Exchange Model (NIEM) are all data models developed through the use of a COI process.

G1144

Develop two-level database models: one level captures the conceptual or logical aspects, and the other level captures the physical aspects.

Rationale:

There are a number of modeling tools available that support entity-relationship diagram (ERD) development. Developers can use these tools to create conceptual/logical models that are independent of the **DBMS** in which the system is implemented and to develop the physical models that are translated directly into data definition language (DDL), the **SQL** code used to create the database. Using a conceptual/logical model permits implementation or reuse of a complex ERD on multiple **DBMS** products.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Have separate **conceptual**/logical and **physical** models been developed?

Procedure:

Verify the presence of a conceptual/logicalmodel0 and a physical model.

Example:

None.

G1146

Include information in the data model necessary to generate a data dictionary.

Rationale:

A **data dictionary** is an integral part of every system including databases. A description of each data item and the units in which the contents are measured are essential. **Data modeling** tools provide a mechanism for storing information necessary to produce a **data dictionary**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the data model include description information?

Procedure:

Examine the physical data model.

Example:

None.

G1147

Use domain analysis to define the constraints on input data validation.

Rationale:

Domain analysis is an integral part of any data system including databases. Domains describe the set or range of values that are acceptable for a specific data item. These include, at a minimum the following:

- Data type
- Precision
- Minimum
- Maximum
- Length

These values are used to validate the data.

In the database, the range checking is done via check constraints on the data item. These **check constraints** are generated from the **physical data model** as part of the DDL.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Does the data model include constraints derived from domain analysis?

Procedure:

Examine the physical data model.

Example:

None.

G1148

Normalize data models.

Rationale:

Normalization is a central **tenet** of **relational database** theory. It is also part of **OOA**.

A database should usually be normalized to at least third normal form. Although there are seven normal forms, normalization beyond third normal form is rarely considered in practical database design.

Objects developed in the absence of data normalization are prone to unnecessary complexity required to keep multiply copies of data.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Is the database design in third normal form?

Procedure:

Examine the conceptual/logical **data model**.

Example:

None

G1151

Define declarative foreign keys for all relationships between tables to enforce referential integrity.

Rationale:

Foreign Key constraints enforce referential integrity. The principle of referential integrity requires that the foreign key values of a child table are either null or match exactly those of the **primary key** in the parent table.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Have foreign-key constraints been incorporated into the database?

Procedure:

Examine the database to determine whether foreign-key constraints have been included in the database creation scripts and created in the database.

Example:

None

G1153

Separate application, presentation, and data tiers.

Rationale:

Separation into tiers allows for the separate maintenance of each tier as long as the interface between tiers does not change. It also allows for multiple implementations of a layer to meet different requirements. This supports technology refresh and certain requirements changes.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the program, project or initiative architecture support clear boundaries between application layers, e.g. data, presentation, and business logic layers.

Procedure:

Examine the program, project or initiative architecture and evaluate the degree to which it supports clear boundaries between applications layers such as data, and presentation layers. Verify that the system design accommodates a multi-tier architecture.

Example:

The use of web services is one means of separating the presentation layer from business logic and data layers.

G1154

Use stored procedures for operations that are focused on the insertion and maintenance of data.

Rationale:

Current software design methodologies and architectures call for the implementation of an n-tiered architecture with business rules in the middle tier and data stored in a separate data tier. When multiple applications access a common database, however, the rules may be best located at the data-tier level. Otherwise, changes in one application would have to be coordinated across all applications.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are database triggers used?

Procedure:

Check for stored procedures that are triggered on insertion, deletion, and update events.

Example:

```
CREATE TRIGGER PersonCheckAge
AFTER INSERT OR UPDATE OF age
ON Person
FOR EACH ROW
BEGIN
  IF (:new.age < 0) THEN
    RAISE_APPLICATION_ERROR
      ( -20000,
        'no negative age allowed'
      );
  END IF;
END;.
```

G1155

Use triggers to enforce referential or data integrity, not to perform complex business logic.

Rationale:

Triggers are fired on events. Current software design methodologies and architectures call for the implementation of an n-tiered architecture with business rules in the middle tier and data stored in a separate data tier. Implementing business logic in triggers, as well as in the middle tier, violates this concept.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Has business logic been incorporated into database triggers?

Procedure:

Examine the database trigger code to determine whether business logic or calls to stored procedures incorporating business logic have been coded into them.

Example:

None

G1190

Use a build tool.

Rationale:

A build tool allows for the encapsulation of building instructions into machine-readable files or sets of files. The instructions can be successfully and consistently repeated.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the program or project use a build tool?

Procedure:

Identify which build tool the program or project is using.

Example:

None.

G1202

Use the CORBA Portable Object Adapter (POA) instead of the Basic Object Adapter (BOA).

Rationale:

The CORBA Basic Object Adapter (BOA) was the CORBA Version 1 specification for the client-server object capability. The BOA specification was found to be so incomplete that vendor-specific interpretations were required for operable implementation. In CORBA Version 2, the Portable Object Adapter (POA) was significantly more complete and flexible. In the current marketplace, POA implementations are standard and, in quality implementations, are not vendor-specific. Consequently, using POA eliminates one significant area of vendor-specific coding.

BOA	***POA***
• Focuses on CORBA server implementations and not CORBA object implementations • Naming convention issues on server side • Tightly coupled to **ORB** implementation • Non-standardized way to connect to ORB • Four activation models for server processes	• Services for lifecycle management • Abstract layer between ORB and object • Standard, portable interface for communicating with ORB runtime • Two servant incarnation styles

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does any CORBA application code reference the **`CORBA::BOA`** identifier.

Procedure:

Review the code for the use of the **`CORBA::BOA`** identifier.

Example:

BOA Coding Example

Client Side

The code below shows a C++ CORBA client BOA initialization for the ORBIX ORB. Other ORB vendors may have different initialization sequences.

```
int main
  ( int argc,
    char **argv
```

```
  )
{ MyServer_var MyVar;
  CORBA::ORB_ptr myOrbPtr
    = CORBA::ORB_init(argc, argv,"Orbix");
  try
  { // The default is the local host:
    MyVar = MyServer::_bind(":ServerName");
  } // End try
  catch ( CORBA::SystemException &sysEx )
  { cerr << "Unexpected system exception" << endl;
    cerr <<&sysEx;
    exit(1);
  } // End CORBA::SystemException
  catch(...)
  { // an error occurred while trying
    // to bind to the grid object.
    cerr << "Bind to object failed" << endl;
    cerr << "Unexpected exception " << endl;
    exit(1);
  } // End catch ...
} // End main
```

Server Side

Use the code below as a model. This example shows a C++ CORBA server BOA init for the ORBIX ORB. For BOA, other ORBS will have a different initialization sequence.

```
try
{ MyObject::myOrb_
    = CORBA::ORB_init(argc, argv, "Orbix");
  MyObject::myboa_
    = MyObject::myOrb_->BOA_init(argc, argv, "Orbix_BOA");
} // End try
catch ( CORBA::SystemException &sysEx )
{ //some exception handling code
} // End catch
try
{ NoeLoggerCfg::myboa_->impl_is_ready("MyServiceName",
  CORBA::ORB::INFINITE_TIMEOUT);
} // End try
catch ( CORBA::SystemException &sysEx )
{ //exception handling code
}
```

POA Coding Example

Client Side

This example shows a C++ CORBA client POA init for the ORBIX ORB. For BOA, other ORBS will have a different initialization sequence.

```
int main
  ( int argc,
    char **argv
  )
{ CORBA::ORB_var myOrb = CORBA::ORB_init(argc, argv);
  try
  { CORBA::Object_var obj
      = ... // however you get the object reference
    if(CORBA::is_nil (obj))
    { cerr << "Nil object reference" << endl;
      throw 0;
    } // End if
  } // End try
  catch ( CORBA::SystemException &sysEx )
  { cerr << "Unexpected system exception" << endl;
    cerr <<&sysEx;
    exit(1);
  } // End catch CORBA::SystemException
```

```
  catch ( ... )
  { cerr << "Unexpected system exception" << endl;
    exit(1);
  } // End catch ...
  myinterface::myobject_var myvar;
  try
  { myvar = myinterface::myobject::_narrow(obj);
  } // End try
  catch ( CORBA::SystemException &sysEx)
  { cerr << "Unexpected system exception" << endl;
    cerr <<&sysEx;
    exit(1);
  } // End catch CORBA::SystemException
} // End main
```

Server Side

Use the code below as a model. This example shows a C++ CORBA server POA init for the ORBIX ORB. For POA, other ORBS will have a different initialization sequence.

```
int main
  ( int argc,
    char *argv[ ]
  )
{ try
  { // initialize the ORB
    orb_var orb  = CORBA::ORB_init(argc, argv, "Orbix");
    // obtain an object reference for the root POA
    object_var obj
      = orb->resolve_initial_references ("RootPOA");
    POA_var poa = POA::_narrow(obj);
    // incarnate a servant
    My_Servant_Impl servant;
    // Implicitly register the servant with the root POA
    obj =  servant._this ();
    //start the POA listening for requests
    poa -> the_POAManager ()->activate ();
    //run the orb's event loop
    orb->run ();
  } // End try
  catch ( CORBA::SystemException &sysEx )
  { // some exception handling code
  } // End catch
} // End main
```

G1203

Localize frequently used CORBA-specific code in modules that multiple applications can use.

Rationale:

In a family of applications, similar patterns of CORBA Object Request Broker (**ORB**) invocation sequences frequently arise. This is common in service object initialization, policy association, discovery, binding, and release handling. Implementing this functionality in a utility library paradigm localizes the code to reduce maintenance and facilitate extensibility, and assures consistency across the family of applications.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Extensibility
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do the standard object policy association CORBA invocations occur in more than one module?

Procedure:

The presence of "**`CORBA::PolicyList`**" in C++ indicates policy presence.

Example:

None

2) Test:

Do the standard object initialization CORBA invocations occur in more than one module?

Procedure:

The presence of "**`CORBA::ORB_var`**" or "**`CORBA::ORB_init`**" in C++ indicates ORB initialization. The presence of "**`CORBA::Object_var`**" in C++ indicates ORB access.

Example:

None

3) Test:

Do the standard object policy association CORBA invocations occur in more than one module?

Procedure:

The presence of "**`CORBA::PolicyList`**" in C++ indicates policy presence.

Example:

None

4) Test:

Do the standard object discovery CORBA invocations occur in more than one module?

Procedure:

The presence of "**`Resolve_NamingService()`**"in C++ indicates intended access to one of CORBA's discovery capabilities.

Example:

None

5) Test:

Do the standard object binding and release CORBA invocations occur in more than one module?

Procedure:

The presence of "**`::_narrow(obj.in())`**" or "**`CORBA::is_nil(`**" in C++ indicates activity associated with obtaining and validating an object binding to a legitimate reference. The presence of "**`CORBA(release)(`**" in C++ indicates intended release of a CORBA-bound object reference.

Example:

None

G1204

Create configuration services to provide distributed user control of the appropriate configuration parameters.

Rationale:

For user-modifiable configuration settings that are intended to be accessible by distributed processes at runtime, the appropriate mechanism for implementation involves **CORBA** services. The first form is a network service to be invoked as a client by the target system application at initialization. This can support a consistent, network-wide distribution of startup parameters. The second form is a service implemented by the target application which allows communication to the application during execution (after startup). This allows **real-time** configuration changes for matters such as Portable Object Adapter (**POA**) instantiation threading policies to address load management.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Decentralized Operations and Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is a service defined in the IDL to obtain the configuration parameters?

Procedure:

Review the code for a service that can be used to obtain configuration.

Example:

The following code is an example of a CORBA server that instantiates a configuration service. The service manages the individual configuration parameters for the servers on the ORB.

Ada Example

```
CORBA.ORB.IIOP_English;
pragma Elaborate_All(CORBA.ORB.IIOP_English);
with CORBA ;
with CORBA.BOA ;
with CORBA.ORB ;
with CORBA.Object ;
with Configuration.Impl ;
with Configuration.Helper ;
with Ada.Exceptions ;
with Ada.Text_IO ;
with my_CORBA ;
with Event_Ada_API ;
procedure Configuration_Server is
    -- required for OrbExpress
    First_Variable : CORBA.ORB.Life_Span ;
    -- declare the object instance
    Configuration_Object : Configuration.Ref ;
    --variables needed for ior writing
    No_Timeout : constant := 0.0;
    Config_Name : constant String
```

```
      := Configuration.Helper.Simple_Name ;
    Config_Host : Corba.String ;
    Config_Port : Corba.String ;
begin -- Configuration_Server
  -- create (and initialize) the object
  -- config file is read and the port needed
  -- is in there
  Configuration_Object
    := Configuration.Impl.Create(Config_Name) ;
  GET_HOSTNAME:
  begin
    Config_Host
      := Configuration.Get_String
         ( Self => Configuration_Object,
           Name => Corba.To_Corba_String
                     ( "Local_Host_Shortname" )
         );
  exception -- GET_HOSTNAME
    when others =>
      Ada.Text_IO.Put_Line
        ( "ERROR: Missing parameter"
         & "<Local_Host_Shortname> "
         & "in the config_parameters.txt file."
        );
  end GET_HOSTNAME;
  GET_CS_PORT:
  begin
    Config_Port
      := Configuration.Get_String
         ( Self => Configuration_Object,
           Name => Corba.To_Corba_String
                     ( "Config_Service_Port" )
         );
  Exception -- GET_CS_PORT
    when others =>
      Ada.Text_IO.Put_Line
        ( "ERROR: Missing parameter "
         & "<Config_Service_Port> "
         & "in the config_parameters.txt file."
        );
  end GET_CS_PORT;
  Ada.Text_IO.Put_Line
    ( "Host => "
        & Corba.To_Standard_String(Config_Host)
        & " Port => "
        & Corba.To_Standard_String(Config_Port)
    );
  --timeout 0 so we can write IOR out
  CORBA.BOA.Impl_Is_Ready
      ( Time_Out             => No_Timeout,
        Server_Instance_Name => Config_Name,
        Listen_On_Endpoints  =>
          "tcp://"
          & Corba.To_Standard_String(Config_Host)
          & ":"
          & Corba.To_Standard_String(Config_Port)
       );
      -- ----------------------------------------------
      -- HERE IS WHERE CODE FOR THE IOR TO BE
      -- USED ON THE C++ ORB
      -- ----------------------------------------------
  -- get the IOR and write it to disk
  my_CORBA.Write_IOR_To_File
    ( Server_Name => Config_Name,
      Server_Ref  =>
        CORBA.Object.Ref(Configuration_Object)
    );
  READY_BLOCK:
  begin
    -- notify subscribers of availability
    -- of configuration parameters via the
    -- event service
    Event_Ada_API.Send
      ( Channel_Name => "Config_Channel",
        Event        => "Configuration Service Ready."
```

```
      );
  Exception - READY_BLOCK
    when others =>
      Ada.Text_IO.Put_line
        ( "Configuration_Server : "
         & Exception sending ready signal."
        );
  end READY_BLOCK;
  Ada.Text_IO.Put_line
    ( "Configuration_Server : "
     & Configuration Service Ready."
    );
  CORBA.BOA.Impl_Is_Ready
    ( Time_Out             => CORBA.Infinite_Timeout,
      Server_Instance_Name => Config_Name
    ) ;
exception -- Configuration_Server
  when X_Other: others =>
    Ada.Text_IO.Put_line
      ( "Configuration_Server : "
       & Ada.Exceptions.Exception_Name(X_Other)
      );
end Configuration_Server ;
```

C++ Example

The following code snippets depict a C++ server that instantiates a version collection service for an About box. It uses the IORs from the servers on the Ada ORB via the IOR files, and invokes those objects to get version information. It uses the utility templates for binding. It exemplifies the approach described in Encapsulate CORBA ORB operations for C++.

Note: *This was done on the ORBIX C++ and Ada ORBs.*

```
#include <iostream.h>
#include <rw/cstring.h>
#ifndef _STDIO_H
#include <stdio.h>
#endif
#ifndef _STRING_H
#include <string.h>
#endif
#ifndef _STDLIB_H
#include <stdlib.h>
#endif
#ifndef _ASSERT_H
#include <assert.h>
#endif
// Include files for all the objects desired for
// collecting version information
//Ada configuration service
#ifndef configuration_hh
#include <configuration.hh>
#endif
// include files for other desired services;
// removed for brevity
// other support objects and utilities
#ifndef _CORBA_UTILS__
#include <corba_utils.h>
#endif
#ifndef __LOG_API_H__
#include <log_api.h>
#endif
#ifndef _VERSION_AGENT_GLOBALS_H_
#include "version_agent_globals.h"
#endif
const RWCString  Version_Agent_i::MSG_VERSION_NOT_FOUND_
  = "Version Info. not found for ";
const CORBA::ULong Version_Agent_i::MAXSERVERS_
```

```
  = 12;
Version_Agent_i:: Version_Agent_i(): theVersionInfoPtr_(0)
{ theVersionInfoPtr_
     = new versionInfoType(MAXSERVERS_);
  theVersionInfoPtr_->length(MAXSERVERS_);
} // End constructor
Version_Agent_i:: ~Version_Agent_i()
{ // Do nothing
} // End destructor
/*********************************************************
FUNCTION NAME: createVersions
PURPOSE: helper function that gets the version info
INPUT:
OUTPUT:
*********************************************************/
void Version_Agent_i::createVersions ()
{ char *iorString;
  int bBindOk = 0;
  int versionCnt = 0;
  versionInfoType* rl = theVersionInfoPtr_;
  CORBA::ULong MAXSERVERS Version_Agent_i::MAXSERVERS_;
  // server variables for all the objects desired
  // for collecting version information
  // most declarations removed for brevity
    EventServiceFactory_var es_var;
  // Ada configuration service
    Configuration_var cfg_var;
  // === load the versions of the individual components
  // Code for other services removed for brevity
  // This is an ADA service using the IOR string
  {  //****************** config service ***************
    logMsg
      ( "get config service version",
        Log_Api::DEBUG_1_MSG
      );
    RWCString errMsg
      ( Version_Agent_i::MSG_VERSION_NOT_FOUND_.data()
      );
    errMsg.append ( "Configuration Service" );
    // here we get the IOR from the ADA orb using
    // the helper methods
      iorString = getIorFile("Configuration");
    //template class to hide binding issues to the ADA ORB
    If ( iorString )
    { Ada_Binder < Configuration,
      Configuration_var > bo ( iorString );
      bBindOk = bo.bindToAda(&cfg_var) ;
      // get the version info and load it
      If ( bBindOk
          && !( CORBA::is_nil(cfg_var))
         )
     { try
       { char* str = cfg_var->version();
         if ( str )
         { (*theVersionInfoPtr_)[versionCnt]
             = CORBA::string_dup(str);
          delete str;
         } // End if
         else
         { (*theVersionInfoPtr_)[versionCnt]
             = CORBA::string_dup(errMsg.data());
         } // End else
       } // End try
       catch(...)
       { (*theVersionInfoPtr_)[versionCnt]
           = CORBA::string_dup(errMsg.data());
       } // End catch
       cfg_var->_closeChannel();
     } // End if
     else
     { (*theVersionInfoPtr_)[versionCnt]
          = CORBA::string_dup(errMsg.data());
     } // End else
     if(iorString)
     { free (iorString);
```

```
        iorString = NULL;
      } // End if
    } //endif iorstring
    else
    { (*theVersionInfoPtr_)[versionCnt]
        = CORBA::string_dup(errMsg.data());
    } // End else
    //leaving scope releases the corba object
  } //end cfg_svf
  bBindOk = 0;
  versionCnt++;
  assert(versionCnt <= MAXSERVERS);
} // End createVersions
/*********************************************************
FUNCTION NAME: start
PURPOSE:  handle startup specific stuff
INPUT:
OUTPUT:
*********************************************************/
void Version_Agent_i:: start
  ( CORBA::Environment &IT_env
  ) throw (CORBA::SystemException)
{ //get all the version info
  createVersions();
} // End start
/*********************************************************
FUNCTION NAME: stop
PURPOSE:  handle stop specific stuff
INPUT:
OUTPUT:
*********************************************************/
void Version_Agent_i:: stop
  ( CORBA::Environment &IT_env
  ) throw (CORBA::SystemException)
{ // Release info
  // Let CORBA time out the service
  logMsg ( "stop received" );
  VersionAgentGlobals::myboa->setNoHangup ( 0 );
  VersionAgentGlobals::myboa->deactivate_impl
    ( "Version_Agent" );
} //end version impl
```

G1205

Use non-source code persistence to store all user-modifiable CORBA service configuration parameters.

Rationale:

For user-modifiable configuration settings that are host-specific and that are not intended to be accessible by distributed processes at runtime, the appropriate mechanism for implementation involves local persistent storage. The appropriate form of local storage depends on the local host architecture and may be file- or host-DBMS oriented. It is important that such parameters are not stored in source code that requires build processes for modification.

For **SOA** services, configuration parameters relating to invoked services should not be service-host-specific at the invoking client application.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are there any user-modifiable configuration parameters hard coded in the non-auto-generated files?

Procedure:

Inspect the code for constant strings or constants that contain configuration parameters.

Example:

None.

G1208

Add new functionality rather than redefining existing interfaces in a manner that brings incompatibility.

Rationale:

By not replacing old methods of objects, library functionality consumers can continue to operate and not be forced to upgrade.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are methods that are being replaced marked with deprecated tags?

Procedure:

Check revision history to make sure that methods are deprecated and not removed unless they have expired. "Expired" means that they have passed the expected shelf life, as defined by the project standards or other standards documentation.

Example:

None

2) Test:

Do new methods being added contain information on methods they are replacing?

Procedure:

Check to make sure newly added methods contain information and rationale on the methods they are replacing.

Example:

None

G1209

For Java, use JDK logging facilities.

Rationale:

Java has a built-in logging framework that is portable across platforms, projects, and installations.

Referenced By:

NESI / Part 5: Developer Guidance / Middleware / Java EE Deployment Descriptors
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the application use anything other than the specified logging frameworks?

Procedure:

Check for use of logging frameworks other than the JDK.

Example:

None

G1210

For .NET, use Debug and Trace from the `System.Diagnostics` namespace.

Rationale:

.NET has a built-in logging framework that is portable across .NET projects and installations.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / .NET Framework
NESI / Part 5: Developer Guidance / Middleware / .NET Framework
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the application use anything other than the specified logging frameworks?

Procedure:

Check for use of logging frameworks other than **`System.Diagnostics`**.

Example:

None

G1213

Provide an architecture design document.

Rationale:

An architectural design document provides evaluators with a roadmap of the application. This helps evaluators verify that the application follows guidance such as using the Model View Controller model.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Do the project deliverables for evaluation include a document that contains the architectural design of the application?

Procedure:

See if an architectural design document exists.

Example:

None

G1214

Provide a document with a plan for deprecating obsolete interfaces.

Rationale:

This information allows users to phase out deprecated interfaces. For instance, Sun plans to maintain backward compatibility for the **JDK** for seven years. This means developers can count on deprecated methods not being removed for seven years.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Do the project deliverables for evaluation include a document that contains a plan for deprecating obsolete interfaces?

Procedure:

See if a document with a plan for deprecating obsolete interfaces exists.

Example:

None.

G1215

Provide a coding standards document.

Rationale:

The standards ensure a consistent code base. A coding standards document defines rules to keep code readable, maintainable, and secure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Secure Coding Standards
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Secure Coding Standards
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Secure Coding Standards
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Do the project deliverables for evaluation include a coding standards document?

Procedure:

See if a coding standards document exists.

Example:

None

G1216

Provide a software release plan document.

Rationale:

The release plan document ensures that there is a formal process for releasing the software. It includes a description of how to acquire the software from the software configuration management (SCM) repository and how to build, label, and release it.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Do the project deliverables for evaluation contain a release plan document?

Procedure:

See if a software release plan exists.

Example:

None

G1217

Develop and use externally configurable components.

Rationale:

To be portable and to accommodate reuse, components must be configurable using external descriptors usually defined in **XML**. Examples of things that might need to be configured include the following:

- A data source for the component to obtain a Java Database Connection (**JDBC**)
- The location of a service with which the component must communicate
- The location of implementation classes that the component uses

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Implement a Component-Based Architecture
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Implement a Component-Based Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Implement a Component-Based Architecture
NESI / Part 5: Developer Guidance / Implement a Component-Based Architecture
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are deployment descriptors used?

Procedure:

Check for the existence of deployment descriptors in the appropriate directories. Usually the file is named **`web.xml`**.

Example:

None

G1218

Use a build tool that supports operation in an automated mode.

Rationale:

During testing, human interaction can be a cause of error and unrepeatable results. Operating in automated mode can eliminate these errors.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have a build all target?

Procedure:

Check the build scripts or descriptors of the build tool for the ability to build the entire project, system, or application.

Example:

None

G1219

Use a build tool that checks out files from configuration control.

Rationale:

To make sure all the parts of the build are under configuration control, compare all files with the configuration baseline, and download the appropriate files.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have a checkout target?

Procedure:

Check the build scripts or descriptors of the build tool for the ability to check out the entire project, system, or application.

Example:

None

G1220

Use a build tool that compiles source code and dependencies that have been modified.

Rationale:

To limit the changes made between builds, only compile code that has been modified. If there are no intermediate files, then compile all files.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have a compile target?

Procedure:

Check the build scripts or descriptors of the build tool for the ability to compile the entire project, system, or application.

Example:

None

2) Test:

Do all the intermediate files (e.g., **`.obj`** or **`.class`**) have the same date and time stamps?

Procedure:

Scan the files for date and time stamps.

Example:

None

G1221

Use a build tool that creates libraries or archives after all required compilations are completed.

Rationale:

Libraries should be able to be recreated independently of any executables and should always verify that any intermediate files are not stale.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have a generate library target?

Procedure:

Check the build scripts or descriptors of the build tool for the ability to generate the composing libraries or archives.

Example:

None

G1222

Use a build tool that creates executables.

Rationale:

An executable is dependent on many files, including source files, intermediate files, and libraries or archives. The building of the executable must support a control process that includes configuration management, compiling, and testing.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have an executable target?

Procedure:

Check the build scripts or build tool descriptors for the ability to build the executables for the entire project, system, or application.

Example:

None

G1223

Use a build tool that is capable of running unit tests.

Rationale:

All code should be able to be tested independently of creating intermediate files, libraries, or executables.

Tests should be unit tests as well as system-level tests.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have a test target?

Procedure:

Check the build scripts or descriptors of the build tool for the ability to test the entire project, system, or application.

Example:

None

G1224

Use a build tool that cleans out intermediate files that can be regenerated.

Rationale:

For security reasons, all files that comprise the build need to be under configuration control. Cleaning out all files is essential in ensuring that only approved code is incorporated into the build.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Does the tool have a clean target?

Procedure:

Check the build scripts or descriptors for the build tool for the ability to remove the entire project, system, or application files.

Example:

None

G1225

Use a build tool that is independent of the Integrated Development Environment.

Rationale:

Some build tools are tightly coupled with an **Integrated Development Environment** (**IDE**) that causes vendor lock-in and license issues when the software is delivered to the Government.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 5: Developer Guidance / Automate the Software Build Process

Evaluation Criteria:

1) Test:

Is the build tool one of the recognized standards, such as ant?

Procedure:

Check for files named **`build.xml`**.

Example:

None

2) Test:

Is the build tool one of the recognized standards, such as **`make`** or **`nmake`**?

Procedure:

Check for files with the name **`makefile`**.

Example:

None

3) Test:

Does the build tool require a license?

Procedure:

Check for files with the name **`makefile`**.

Example:

None

G1237

Do not hard-code the configuration data of a Web service vendor.

Rationale:

Some vendors generate code that passes Web service vendor-specific configuration data during initialization or startup. This reduces the portability of the code and can cause maintenance problems later.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Web Services / Insulation and Structure
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Web Services / Insulation and Structure
NESI / Part 5: Developer Guidance / Middleware / Web Services / Insulation and Structure
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is there any Web service vendor-specific configuration data in the client code?

Procedure:

Parse the code and look for hard-coded configuration data that might be used to configure the vendor's Web service.

Example:

None

G1239

Use design patterns (e.g., facade, proxy, or adapter) or property files to isolate vendor-specifics of vendor-dependent connections to the enterprise.

Rationale:

This isolation increases maintainability. Guidance G1071 asserts that vendor-neutral connection mechanisms should be used. When vendor-specific connection mechanisms are unavoidable, this guidance will apply.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Is the connection mechanism vendor-dependent?

Procedure:

Examine the source code for vendor-specific imports or includes.

Make sure that all references to the vendor-specific connection mechanisms are isolated to a single class (like a helper) or set of methods that are used as part of an isolation design pattern such as facade, proxy, or adapter.

Also, look for hard-coded vendor-specific connection strings.

Example:

None

G1245

Isolate the Web service portlet from platform dependencies using the Web Services for Remote Portlets (WSRP) Specification protocol.

Rationale:

The **OASISWSRP** 1.0 Specification accounts for the fact that producers and consumers may be implemented on very different platforms, such as a Java EE-based Web service, a Web service implemented on the Microsoft .Net platform, or a **portlet** published directly by a **portal**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Web Portals
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Web Portals
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Decentralized Operations and Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the Web service implement the WSRP Registration interface?

Procedure:

Look for the occurrence of the **`getService`**, **`register`**, **`deregister`**, and **`modifyRegistration`** methods as defined in the OASIS WSRP Specification.

Example:

```
public static RegistrationService getService
   ( java.lang.String baseEndpoint
   ) throws java.lang.Exception
public RegistrationContext register
  ( java.lang.String consumerName,
    java.lang.String consumerAgent,
    boolean methodGetSupported,
    java.lang.String[] consumerModes,
    java.lang.String[] consumerWindowStates,
    java.lang.String[] consumerUserScopes,
    java.lang.String[] customUserProfileData,
    Property[] registrationProperties
  ) throws java.lang.Exception
public ReturnAny deregister
  ( java.lang.String registrationHandle,
    byte[] registrationState
  ) throws java.lang.Exception
public RegistrationState modifyRegistration
  ( RegistrationContext registrationContext,
    RegistrationData registrationData
  ) throws java.lang.Exception
```

2) Test:

Does the Web service implement the WSRP Service Description interface?

Procedure:

Look for the occurrence of the **`getService`**, **`register`**, and **`getServiceDescription`** methods as defined in the OASIS WSRP Service Description API Specification.

Example:

```
public static ServiceDescriptionService getService
  ( java.lang.String baseEndpoint
  ) throws java.lang.ExceptionThrows:
jpublic ServiceDescription getServiceDescription
  ( RegistrationContext registrationContext,
    java.lang.String[] desiredLocales
  ) throws java.lang.Exception
```

3) Test:

Does the Web service implement the WSRP Portlet Configuration interface?

Procedure:

Look for the occurrence of the **`getService`**, **`getPortletDescription`**, **`clonePortlet`**, **`destroyPortlets`**, **`setPortletProperties`**, **`getPortletProperties`** and **`getPortletPropertyDescription`** methods as defined in the OASIS WSRP Portlet Configuration API Specification.

Example:

```
public static PortletManagementService getService
  ( java.lang.String baseEndpoint
  ) throws java.lang.Exception
public PortletDescriptionResponse getPortletDescription
  ( RegistrationContext registrationContext,
    PortletContext portletContext,
    UserContext userContext,
    java.lang.String[] desiredLocales
  ) throws java.lang.Exception
public PortletContext clonePortlet
  ( RegistrationContext registrationContext,
    PortletContext portletContext,
    UserContext userContext
  ) throws java.lang.Exception
public DestroyPortletsResponse destroyPortlets
  ( RegistrationContext registrationContext,
    java.lang.String[] portletHandles
  ) throws java.lang.Exception
public PortletContext setPortletProperties
  ( RegistrationContext registrationContext,
    PortletContext portletContext,
    UserContext userContext,
    PropertyList propertyList
  ) throws java.lang.Exception
public PropertyList getPortletProperties
  ( RegistrationContext registrationContext,
    PortletContext portletContext,
    UserContext userContext,
    java.lang.String[] names
  ) throws java.lang.Exception
public PortletPropertyDescriptionResponse getPortletPropertyDescription
  ( RegistrationContext registrationContext,
    PortletContext portletContext,
    UserContext userContext,
    java.lang.String[] desiredLocales
  ) throws java.lang.ExceptionThrows
```

4) Test:

Does the Web service implement the WSRP Markup interface?

Procedure:

Look for the definition of the **`getMarkup`**, **`performBlockingInteraction`**, **`initCookie`** and **`releaseSessions`** methods as defined in the OASIS WSRP Markup API Specification.

Example:

```
public MarkupResponse getMarkup
   ( RegistrationContext registrationContext,
     PortletContext portletContext,
     RuntimeContext runtimeContext,
     UserContext userContext,
     MarkupParams markupParams
   ) throws java.lang.Exception
public void performBlockingInteraction
   ( RegistrationContext registrationContext,
     PortletContext portletContext,
     RuntimeContext runtimeContext,
     UserContext userContext,
     MarkupParams markupParams,
     InteractionParams interactionParams
   ) throws java.lang.Exception
public Extension[] initCookie
   ( RegistrationContext registrationContext
   )  throws java.lang.Exception
public Extension[] releaseSessions
   ( RegistrationContext registrationContext,
     java.lang.String[] sessionIDs
   ) throws java.lang.Exception
```

G1267

Use industry standard HTML data entry fields on Web pages.

Rationale:

Macromedia Flash and Java Applets can also be used for data input but are not HTML standards and tend to decrease the maintainability of a Web site.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do any Web pages have data entry fields?

Procedure:

Search all Web pages for the "applet" and "embed" tags. Load each page found in the search by loading and visually inspecting to see if Flash or Applets are used for data entry.

Example:

Correct Usage:

Incorrect usage:

Applet	
Flash	

G1268

Label all data entry fields.

Rationale:

A label provides the user with a brief description of the text to be entered. Labels are essential for a user to understand the data entry field.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are all data entry fields labeled?

Procedure:

Search all Web pages for the word "form" and load each resulting Web page in a browser. Visually inspect each data entry field to make sure it has labels.

Example:

None.

G1270

Include scroll bars for text entry areas if the data buffer is greater than the viewable area.

Rationale:

Scroll bars provide a visual cue to the user that the text extends beyond the viewable area. Scroll bars will appear by default for an HTML text area.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do any Web pages turn off scroll bars for text areas?

Procedure:

Search all Web pages and style sheets for the phrase "overflow:hidden" or a form thereof. This turns off scroll bars using styles, but only works in certain browsers. Make sure it is not used.

Example:

Correct Usage

Scroll bars should not be hidden.

Incorrect Usage

Inline style:

```
<html>
<body>
<form>
<textarea style="overflow:hidden"></textarea>
</form>
</body>
</html>
```

External style:

```
textarea.scroll {
   overflow:hidden;
}
```

G1271

Provide instructions and HTML examples for all style sheets.

Rationale:

An instruction manual will enable developers to use the style sheet correctly and efficiently.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Extensibility
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are instructions included for each style sheet provided?

Procedure:

Verify that a document is provided that contains instructions and example code for each style provided.

Example:

Correct usage:

```
Cascading style sheet:
.td-items {
   text-align:right;
}
```

Example of usage:

Incorrect usage:
No HTML example explaining style usage.

G1276

Do not modify the contents of the Web browser's status bar.

Rationale:

Using the browser's status bar to display text unrelated to status affects interoperability because a user expects the status bar to provide status and nothing else.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do any of the Web pages modify the browser status bar?

Procedure:

Search every Web page for the word "status" and visually inspect each of the search results to see if the status bar has been modified.

Example:

Correct usage:

```
Web pages contain no references to window.status
```

Incorrect usage:

```
window.status = 'text to display in status bar'
```

G1277

Do not use tickers on a Web site.

Rationale:

Tickers can irritate the user and use unnecessary bandwidth.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do any Web pages contain scrolling text?

Procedure:

Most tickers are written using Applets or Flash. Search all Web pages for the "applet" and "embed" tags. Load each page found in the search and visually inspect to make sure no tickers exist.

Example:

Correct usage:

```
No applet or flash references contain tickers.
```

Incorrect usage:

Applet:
```
applet code="myticker.class" width="200" height="200"
```
Flash:
```
embed src="myticker.swf" width="200" height="200"
```

G1278

Use the browser default setting for links.

Rationale:

Browsers underline links by default. Do not rely on "mouse over" to identify links. Using mouse over to designate links can confuse and slow down infrequent users because they are uncertain which links perform which functions.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do any Web pages or style sheets modify the browser default settings for links?

Procedure:

Search all the Web pages and style sheets for "A:link," "A:visited" and "A:active." Inspect all search results and make sure none of them modify the "A:" items.

Example:

Correct usage:

```
Web pages and style sheets should have no reference to A:link, A:visited or A:active.
```

Incorrect usage:

```
A:link, A:visited, A:active {
   text-decoration:none;
}
```

G1283

Use linked style sheets rather than embedded styles.

Rationale:

Only by referencing an external file will you be able to update the look of an entire Web site with a single change. Also, by pulling style definitions out of the pages, they (Web pages) will be smaller and faster to download.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does a Web page use the LINK tag to include external style sheets instead of embedding styles?

Procedure:

View the source of the HTML page. The header tag (head) should contain links to external style sheet (.css) files. The header tag should not contain any style tags.

Example:

Correct usage:

External style:

```
<head>
  <link rel=stylesheet href="style.css" type="text/css" media=screen>
  <link rel=stylesheet href="basic.css" type="text/css" media=screen>
</head>
```

Incorrect usage:

Embedded style:

```
<head>
  <style type="text/css">
      td {
      background:#ff0;
      }
  </style>
</head>
```

G1284

Use only one font for HTML body text.

Rationale:

Users may not have a wide variety of fonts available in their browser, so it is best to use a single, common font. The general standard is to make body text sans serif since most people find sans serif fonts easier to read on monitors and **serif** fonts better for printed materials.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the HTML or style sheet refrain from using more than one font?

Procedure:

Search all Web pages and style sheets for the word "font." Make sure only one type of font is used for body text. May need to visually inspect Web pages to see if a defined font style is used within the body.

Example:

Correct usage:

Cascading style sheet:

```
body.main {
   font:sans-serif;
}
```

HTML:

Incorrect usage:

Several font styles are used within a body.

G1285

Use relative font sizes.

Rationale:

Relative font sizes make Web sites more accessible and support meeting the requirements of Section 508 of the Rehabilitation Act of 1973. Relative font sizes allow for a low-vision user to enlarge the size of the text. Relative font sizes also support maintainability by not hard coding fixed **font sizes**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are any absolute font sizes utilized?

Procedure:

Search all Web pages and style sheets for the word "font." Inspect the results to make sure no fixed fonts are used (e.g., 12pt).

Example:

Correct Usage

Relative or no font sizes settings are used.
Cascading style sheets:

```
p {
   font-size:200%;
}
p {
   font-size:2em;
}
```

Incorrect Usage

Cascading style sheets:

```
p {
   font-size:12pt;
}
```

HTML (the font attribute should not be used at all within HTML code, only external style sheets):

G1286

Provide text labels for all buttons.

Rationale:

Users need to understand the purpose of all buttons. In some cases an image on the button is not sufficient to convey meaning. Screen scrapers used by the visually impaired work better when text labels are available for buttons

In cases where icons serve as buttons in order to fit within a small display device (such as a personal digital assistant), providing an option to enable text labels (or providing alternate attributes in the case of Web-based interfaces) supports screen scrapers.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do all buttons have associated text labels?

Procedure:

Inspect the user interface to verify text labels are available for all buttons.
Text labels may optionally be displayed:
- on or near the button
- as a tooltip when the user hovers over a button
- as part of a help system where a user clicks and identify tool and then clicks a button.

Button label text may not be enabled by default on all applications, especially systems with small resolution screens such as PDAs.

Example:

Correct usage:

```
<form action="mailto:me@abc.com"
method="post">
<input type="submit" name="emailbut"
value="Send feedback" />
</form>
```

Incorrect usage (using images only):

```
<input type="image" src="send.gif" name="
emailbut"/>
```

G1287

Provide feedback when a transaction will require the user to wait.

Rationale:

Users may think that the application has stopped running or is malfunctioning.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the application provide feedback during long processes?

Procedure:

Run the application and observe any processes that take longer than 10 seconds to complete. Observe if any status indication is provided to alert the user of the status.

Example:

None

G1292

Use text-based Web site navigation.

Rationale:

Text-based navigation works better than image-based navigation because it enables users to understand the link destinations. Users with text-only browsers and browsers with deactivated graphics can see only text-based navigation options.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are there any instances where graphics are used for navigation?

Procedure:

Visually inspect all Web pages and make sure navigation elements are textual.

Example:

None

G1294

Provide a site map on all Web sites.

Rationale:

A site map shows explicit organization of the site. Inexperienced users do not readily form a mental model of the way that information is organized in a Web site, making it hard for them to recover from navigational errors.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Web site have a site map?

Procedure:

Search all Web pages for anything with the name "sitemap," "site map" and "map." Visually inspect the search results to make sure a site map is included.

Example:

None

G1295

Provide redundant text links for images within an HTML page.

Rationale:

Redundant text links for images within an **HTML** page allow users to navigate the **Web page** even if their browsers do not display images (as in situations where the **Web browser** renders content without images due to bandwidth considerations). Screen scrapers that assist the visually impaired also use redundant text links. Images may occur within Web pages as part of the content or navigation controls to include **image maps**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are alternative text links provided for all HTML page images used for navigation?

Procedure:

Verify that alternative text links are provided for images used for navigation by inspecting the HTML source code and testing the HTML page in a browser with image rendering turned off.

Example:

None.

G1300

Secure all endpoints.

Rationale:

Something is only as secure as its weakest link. Therefore, all access points in an application should be secured. An endpoint is defined as an entry or an exit point of an application. Any access point can be vulnerable to attacks. For instance, if an application file reads configuration settings from a properties file, that file can be corrupted or incorrectly configured. This can cause incorrect behavior in the application. Also if component, **module** or application provides remote access or is part of any inter-process communications, these areas are vulnerable to attacks. For instance, if the application provides an external socket interface, does it validate commands being sent by the client?

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity

Evaluation Criteria:

1) Test:

Does the application handle invalid configuration, provide appropriate defaults, and protect sensitive data?

Procedure:

Check application processing of data files (configuration files, properties files, preferences, XML, etc.).

Example:

None.

2) Test:

Does the application properly handle security when dealing with externally accessible API(s) and external ports?

Procedure:

Verify sensitive data is protected, and verify all network base protocols validate commands and values.

Example:

None.

G1301

Practice layered security.

Rationale:

An application with layered security provides more protection against attacks. Combining multiple layers of security defenses can provide additional protection when one layer is broken.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES Definitions and Status
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES Definitions and Status
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Layering and Modularity
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Practice Defense in Depth
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Practice Defense in Depth
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Practice Defense in Depth

Evaluation Criteria:

1) Test:

Do internal and external API(s) perform security checks?

Procedure:

Make sure layers of API(s) starting from externally accessible API(s) down through the layers of internally accessible API(s) provide sufficient security checks. For example, does each layer of the API perform data validation? If internal API is calling remote services, is the data sufficiently protected from snoopers (e.g., use of secure sockets)?

Example:

None

2) Test:

Does the application handle security when processing data files?

Procedure:

Embed all application specific resources such as graphics, internal application configuration files such as internationalization properties/resources, XML files as part of a signed application deployment file (.jar, .exe, etc.).

Example:

None

G1302

Validate all inputs.

Rationale:

Do not limit input validation to the presentation tier; rather, all external APIs should validate inputs prior to use. This is just one aspect of defense in depth which can prevent many attacks including SQL Injection, Cross-Site Scripting, Buffer Overflows, and Denial of Service.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Data, Application and Service Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Data, Application and Service Integrity
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the application provide proper handling for null input?

Procedure:

Check application handling of null values.

Example:

None

2) Test:

Does the application use prefix or postfix validation (asserts) to verify input parameters?

Procedure:

Check application range validation of externally accessible API(s).

Example:

None

G1304

Unit test all code.

Rationale:

A high percentage of all security violations can be attributed to inadequate or non-existent unit testing. Hackers can take advantage of these.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Quality Assurance to Software Development
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Quality Assurance to Software Development
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Quality Assurance to Software Development
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the project unit test the code base?

Procedure:

Use a coverage tool to determine how much of the project's code have been tested.

Check for use of a unit testing framework (JUnit for example).

Example:

None

G1306

Identify and authenticate users of the application.

Rationale:

This ensure there is some traceability and also provides the first in a multilayer security system.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Identity Management / Public Key Infrastructure
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Identity Management / Public Key Infrastructure
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Authorization and Access Control
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Authorization and Access Control

Evaluation Criteria:

1) Test:

Does the application authenticate with another service (**LDAP**, database or simple password)?

Procedure:

Inspect application code to ensure that the user is authenticated against an LDAP, database or simple password service.

Example:

None

2) Test:

Does the application require user certificates?

Procedure:

Ensure the application is setup to require client side certificates. This can be done easily by using a machine without any DoD client certificates installed and attempting to access the application.

Example:

None

G1308

Configure Public Key Enabled applications to use a Federal Information Processing Standard (FIPS) 140-2 certified cryptographic module.

Rationale:

The guidance defines the application types required to support DoD class 3 PKI.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications

Evaluation Criteria:

1) Test:

Is the application using an approved **Federal Information Processing Standard** (**FIPS**) 140-2 cryptographic **module**?

Procedure:

Check the cryptographic module to see if it is FIPS 140-2 compliant.

Example:

None

G1309

Make applications handling high value unclassified information in Minimally Protected environments Public Key Enabled to interoperate with DoD High Assurance .

Rationale:

This guidance defines the application types required to support DoD High Assurance (Mission Assurance Category I [MAC I]) certificates.
The definition of MAC I is "systems handling information that is determined to be vital to the operational readiness or mission effectiveness of deployed and contingency forces in terms of both content and timeliness. The consequences of loss of integrity or availability of a MAC I system are unacceptable and could include the immediate and sustained loss of mission effectiveness. MAC I systems require the most stringent protection measures." (DoD Instruction 8580.1, ***Information Assurance (IA) in the Defense Acquisition System***, 9 July 2004. [R1199])

Note: *This guidance is derived from DoD Instruction 8520.2,* ***Public Key Infrastructure (PKI) and Public Key (PK) Enabling****, 1 April 2004.* *[R1206]*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications

Evaluation Criteria:

1) Test:

Is the application using a High Assurance key material generated in a **Federal Information Processing Standard** (**FIPS**) 140 Level 2 validated hardware cryptographic **module**?

Procedure:

Check cryptographic module to see if it is FIPS 140 Level 2 compliant.

Example:

None.

G1310

Protect application cryptographic objects and functions from tampering.

Rationale:

If cryptographic objects such as private keys, key store, and CA trusted certificates are not protected, the system is not secure.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications

Evaluation Criteria:

1) Test:

Are cryptographic objects protected?

Procedure:

Check that key stores, private keys, and **trust points** are protected.

Verify a documented procedure for creating and documenting the creation of keys exists.
Verify a documented procedure for obtaining certificates exists.
Verify a documented procedure for backing up cryptographic objects exists.

Example:

Use High Security Level setting in Internet Explorer to ensure password protection is used. See https://infosec.navy.mil/PKI/certs.html for software certificate steps. See https://infosec.navy.mil/PKI/cac.html for CAC.

G1311

Use Hypertext Transfer Protocol over Secure Socket Layer (HTTPS) when applications communicate with DoD Public Key Infrastructure (PKI) components.

Rationale:

These are the DoD approved protocols and the only supported ones.

> ***Note:*** *This guidance is derived from DoD Instruction 8520.2,* ***Public Key Infrastructure (PKI) and Public Key (PK) Enabling****, 1 April 2004.* *[R1206]*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications

Evaluation Criteria:

1) Test:

Does the application use only HTTPS to communicate when using DoD PKI?

Procedure:

Have application access the DoD PKI Global Directory Service (GDS) Directory (dod411.gds.disa.mil/) via HTTPS.

Example:

None

G1312

Make applications capable of being configured for use with DoD PKI.

Rationale:

Applications must be configurable to request and install certificates, add **trust points**, and require client authentication.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Section 4.4, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications

Evaluation Criteria:

1) Test:

Is there a capability to configure the application for use with DoD PKI?

Procedure:

Check to make sure the application is configurable to accept certificates, load key stores, and add **trust points**; this may involve inspecting user and administrator manuals.

Example:

None

G1313

Provide documentation for application configuration for use with DoD PKI.

Rationale:

Correct configuration is required for ensuring security. Without detailed documentation, personnel with limited knowledge of security or PKI will have little chance of keeping the overall system secure. The Navy Public Key Infrastructure training site, https://infosec.navy.mil/PKI/training.html (DoD PKI Certificate required for access), contains links to several configuration guides.

Note: *This guidance is derived from the DoD Instruction 8520.2,* ***Public Key Infrastructure (PKI) and Public Key (PK) Enabling****, 1 April 2004.* *[R1206]*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Public Key Infrastructure (PKI) and PK Enable Applications

Evaluation Criteria:

1) Test:

Is there documentation (such as Standard Operating Procedures [SOPs]) on how to configure and setup the application to interoperate within the DoD PKI?

Procedure:

Verify by inspection of the SOPs and by a demonstration that the application performs as documented when the configuration guidance is followed.

Example:

Most application manuals have detailed instructions in enabling PKI (either under the heading "enabling SSL" or "certificates").

G1314

Provide applications the ability to import and export keys (software certificates only).

Rationale:

The whole PKI system is predicated on the use of public-private key pairs. The ability to import (recover) and export (backup) key pairs is critical to a functional PKI application.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Section 4.5, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the application able to import and export its key pair for backup/recovery purposes?

Procedure:

Have the application import and export a key pair.

Note: *Verify the correctness of the exported file through analysis.*

Example:

Internet Explorer can import/export certificates using Tools > Internet Options. Click on Internet tab and then click on Certificates link. Import/Export options are located here.

UNIX-based Web server keys are exported by making a copy of the keys file and placing it in a safe location.

G1316

Ensure that applications protect private keys.

Rationale:

In order for the PKI system to stay secure, the private key must not be compromised. Protecting the private key helps prevent attackers from decrypting secured data communications.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Section 4.5, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the application use and store the private key securely?

Procedure:

Check for the following:

- all copies of the private key destroyed when private key operation is complete; for example, check that the private key does not stay in application memory permanently

- the private key is password protected with a strong password
- the **keystore** is password protected with a strong password

Example:

Attempt to view the contents of the private key using a document viewer program.

G1317

Ensure applications store Certificates for subscribers (the owner of the Public Key contained in the Certificate) when used in the context of signed and/or encrypted email.

Rationale:

This will allow other parties to use the public key to encrypt messages sent to the application.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document. Section (4.5), Version 1.0, July 13, 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the public key available from the Directory Server application?

Procedure:

See if it is possible to extract the public key certificate from the Directory Server application.

Example:

None

G1318

Develop applications such that they provide the capability to manage and store trust points (Certificate Authority Public Key Certificates).

Rationale:

This will ensure the certificate is valid and expedite verification of the certificate.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the Certificate Authority public key available from the application?

Procedure:

View the application's trust list to verify DoD PKI Class 3 CA certificates are present.

Example:

For Internet Explorer, view the DoD PKI Class 3 CA certificates by selecting **`Tools`**>**`Internet Options`**. Click on the **`Internet`** tab and then click on the **`Publishers`** button. Click on the **`Trusted Root Certification Authorities`** tab and scroll down to verify that the DoD PKI Class 3 CA certificates are present.

Web server Certificate Authority certificates can usually be viewed by the application's GUI. If a GUI is not offered, reference the application's manual concerning certificate management.

G1319

Ensure applications can recover data encrypted with legacy keys provided by the DoD PKI Key Recovery Manager (KRM).

Rationale:

Applications may have the need to decrypt legacy information that the application originally encrypted.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Key Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the application able to recover legacy encrypted data?

Procedure:

Acquire the legacy key and demonstrate the ability
to decrypt data that is encoded by that key.

Example:

None

G1320

Use a minimum of 128 bits for symmetric keys.

Rationale:

Strong encryption helps to prevent unauthorized data decryption using modern day resources.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Are symmetric key encryption levels at least 128 bit?

Procedure:

Check the server configuration and verify that the symmetric keys being used are at least 128 bit.

Example:

Verified Web server ciphers under the SSL portion of the configuration pages of the administration server.

For Internet Explorer 5.0 and above, click the **`Help`** menu and then click the **`About Internet Explorer`** option. The About box will list the Cipher Strength.

2) Test:

Is the application using domestic (U.S.) grade ciphers?

Procedure:

Verify that the application supports domestic (U.S.) grade ciphers.

Example:

None.

G1321

Enable applications to be capable of performing Public Key operations necessary to verify signatures on DoD PKI signed objects.

Rationale:

An application must verify the digital signature and check its validity against the current **Certificate Revocation List** (**CRL**) maintained by an on-line repository (e.g., **Online Status Check Responder** or **OSCR**).

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Does the application verify signed objects?

Procedure:

Check that the application validates signed objects against DoD root certificates.

Check that the signing certificate has not been revoked by checking against Certificate Revocation Lists or using the Online Certificate Status Protocol (OCSP).

Example:

Make a back-up copy of the certificate. For Windows based applications, stop the application and edit the signature of the certificate and save the certificate. Start the application back up. The application should fail to start as the signature check will fail.

For validity checking, confirm a validity check of the certificate was performed by viewing the application's audit log.

G1322

Ensure that applications that interact with the DoD PKI using SSL (i.e., HTTPS) are capable of performing cryptologic operations using the Triple Data Encryption Algorithm (TDEA).

Rationale:

Applications must use cryptographic modules approved under **Federal Information Processing Standard** (**FIPS**) 140, Level 1.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Mediate Security Assertions

Evaluation Criteria:

1) Test:

Does the application use TDEA for encrypting and decrypting data?

Procedure:

Inspect the application's configuration file to confirm that TDEA is used for encrypting and decrypting data.

Example:

Most server based applications have cipher related information stored under SSL, certificates, or security. Verify that the application is using TDEA.

G1323

Generate random symmetric encryption keys when using symmetric encryption.

Rationale:

If the application can not generate random keys, then it is vulnerable to attacks if attackers can determine the algorithm for generating the random symmetric encryption keys.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Does the application generate random symmetric encryption keys?

Procedure:

Verify that the random seed is generated (e.g., by viewing the application's vendor documentation).

Example:

Most server based applications either user MOD_SSL or OPEN_SSL. These two toolkits properly use random seed generators.

Apache based servers may require the administrator to type random keystrokes on the keyboard. This process is generating the random seed.

G1324

Protect symmetric keys for the life of their use.

Rationale:

Symmetric key encryption algorithms are based on trivially related keys for both encryption and decryption. The advantage of symmetric key encryption is that it is much less computationally intensive for encryption and decryption compared to asymmetric algorithms. The disadvantage is that the shared symmetric key must be kept secure during storage and transmission.

To prevent disclosure, new symmetric keys are often generated for each unique **session** and exchanged using another encryption algorithm. Store symmetric keys that are used long term carefully to prevent disclosure.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Are symmetric keys stored in unprotected locations?

Procedure:

Check for hard coded symmetric keys in source code or files with weak permissions.

Example:

Symmetric keys should be generated for each session and destroyed when the session is destroyed, never stored in a file with weak permissions or hard coded in source code.

G1325

Encrypt symmetric keys when not in use.

Rationale:

Symmetric keys enable both sides of the conversation to have knowledge of the key for encryption. It can not be given out freely, which means if it is going to be stored for repeated use, it should be encrypted first before storage.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Does the application encrypt symmetric keys when not in use?

Procedure:

Check that the application encrypts symmetric keys during storage.

Example:

None.

G1326

Ensure applications are capable of producing Secure Hash Algorithm (SHA) digests of messages to support verification of DoD PKI signed objects.

Rationale:

Symmetric keys enable both sides of the conversation to have knowledge of the key for encryption. It can not be given out freely, which means if it is going to be stored for repeated use, it should be encrypted first before storage.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Does the application use SHA digest?

Procedure:

Visually validate that the SHA digest is used for symmetric keys.

Example:

Most application servers allow one to configure the hash to SHA1. Please note that the default for most applications is MD5.

G1327

Enable an application to obtain new Certificates for subscribers.

Rationale:

If the application generates subscriber keys, the application shall demonstrate the ability to generate keys, request new certificates, and obtain new certificates through interaction with the DoD PKI. If the generated keys are for encryption applications, the application shall demonstrate its ability to provide keys to the DoD PKI KRM.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Section 4.3.2.2, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Can the application request and obtain new certificates for subscribers?

Procedure:

For application servers, verify that the application can successfully request a certificate via the appropriate certificate request page from a DoD PKI CA.

For application servers, verify that the application can successfully download an issued certificate from a DoD PKI CA.

Example:

Instructions in obtaining a DoD PKI certificate for a user are available at https://infosec.navy.mil/PKI/users.html.

Instructions for obtaining a DoD PKI certificate for web servers including Netscape, Lotus, and IIS is available at https://infosec.navy.mil/PKI/training.html.

G1328

Enable an application to retrieve Certificates for use, including relying party operations.

Rationale:

The ability to retrieve certificates from DoD certificate repositories further ensures the authenticity of the certificate .

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Section 4.3.2.3, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Can the application retrieve **Certificates** from a DoD PKI certificate repository?

Procedure:

Verify that the application can communicate with a DoD PKI certificate repository such as GDS.

Example:

This test procedure is only required for applications that must send encrypted e-mail. For this scenario, assume that Outlook is used; instructions for using Outlook 2000 are available at https://infosec.navy.mil/PKI/Outlook_2000_0704.pdf

G1330

Ensure applications are capable of checking the status of Certificates using a Certificate Revocation List (CRL) if not able to use the Online Certificate Status Protocol (OCSP).

Rationale:

Applications must verify the validity of the certificate prior to establishing trust with another entity. **CRL** is the legacy mechanism for validating certificates. Applications should favor **OSCP** for new development.

Applications operating in environments with network connectivity to a **CRL distribution point** should be able to obtain a current CRL. Applications should be able, without user intervention, to obtain a current CRL to check the status of a certificate that contains a CRL distribution point extension. Applications with network connectivity unable to find CRL distribution points automatically should be capable of being configured with a distribution point that the application then uses to obtain CRLs as needed.

Systems on DoD networks must use a local Web cache to obtain the latest DoD PKI issued CRL per Joint Task Force Global Network Operations (JTF GNO) Communications Tasking Order (CTO) 07-015 of 11 December 2007 (specifically Task 11; DoD PKI Certificate required for access). Configuration instructions for known Web cache products in use and alternative CRL caching capabilities are available from the following location: https://www.us.army.mil/suite/page/474113 (Army or Defense On Line [AKO or DKO] site registration and DoD PKI Certificate required for access).

Note: *This guidance is derived from DoD Instruction 8520.2,* ***Public Key Infrastructure (PKI) and Public Key (PK) Enabling****, 1 April 2004.* *[R1206]*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Can the application perform Certificate status checking with a CRL?

Procedure:

Verify that the application can download a CRL successfully .

Example:

Visually inspect the application is configured to use CRLs for validity checking. This can be achieved by looking at the directory in which the application stores the CRLs.

G1331

Ensure applications are able to check the status of a Certificate using the Online Certificate Status Protocol (OCSP).

Rationale:

Applications must verify the validity of the certificate prior to establishing trust with another entity. CRL is the legacy mechanism for validating certificates. Applications should favor **OCSP** for new development.

Applications may use an OSC responder to check the status of a particular certificate when the DoD has an operational responder. Applications shall prepare and transmit the request to the responder using HTTP in accordance with the DoD Class 3 PKI Infrastructure Interface Specification.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Section 4.3.2.4.2, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Can the application perform **Certificate** status checking with **OCSP**?

Procedure:

Verify that the application can performing OCSP queries to an **OSC** Responder successfully.

Example:

Visually inspect the application is configured to use OCSP for validity checking. This can be achieved by looking at the configuration file to see that the application is configured to use OCSP. One can also visually look at the application's log file to validate that the application is making OCSP queries.

G1333

Only use a Certificate during the Certificate's validity range, as bounded by the Certificate's "Validity - Not Before" and "Validity - Not After" date fields.

Rationale:

Expired certificates should not be accepted except in cases where legacy data was archived.

> ***Note:*** *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Do the date and time of the use of the Certificate fall within the Certificate's validity period?

Procedure:

Visually inspect the certificate's validity dates. The certificate should be valid and not expired.

Example:

Each digital certificate has a lifetime. When viewing a certificate, the certificate will have a valid from date and a valid to date. The current date should fall within this range.

G1335

Make applications capable of being configured to operate only with PKI Certificate Authorities specifically approved by the application's owner/managing entity.

Rationale:

Using approved PKI Certificate Authorities ensures certificate authenticity and ensures that the certificate is chained to the issuer. DoD trust points ensure certificates are chained to the issuer of the certificate and are authentic.

For example, DoD applications are configured to use DoD PKI Certificate Authorities only per the DoD Class 3 PKI - Public Key-Enabled Application Requirements Document Version 1.0, 13 July 2000.

Note: *This guidance is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the application configured to operate only with approved PKI Certificate Authorities?

Procedure:

Visually inspect that only the DoD PKI certificates are trusted by the application.

Example:

Applications typically allow one to view the trust points via the administrative interface to the application. CA certificates are typically located under Certificate Management, SSL, or Security.

G1338

Ensure that Public Key Enabled applications support multiple organizational units.

Rationale:

DoD requirements dictate that certificates shall support multiple organizational units.

> ***Note:*** *This guidance is derived from DoD Instruction 8520.2, Public Key Infrastructure (PKI) and Public Key (PK) Enabling, 1 April 2004.*[R1206]

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Certificate Processing
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Can the application process a **Certificate** that contains multiple organizational units in the Distinguished Name?

Procedure:

Visually inspect the DoD PKI CA certificates stored in the application. You will notice that each certificate contains multiple organizational units (OU=DoD, OU=PKI)

Example:

The majority of certificate request forms do not contain entries for multiple organizational units. In this case, include all of the organizational unit information in the single line. For example, for Navy, please enter the following information next to the Organizational Unit line: Navy, OU=DoD, OU=PKI.

Once the certificate is issued, visually inspect this certificate to verify that the certificate contains these Organizational Unit values.

G1339

Practice defensive programming by checking all method arguments.

Rationale:

Data validation is not limited to Graphical User Interfaces. API(s) and library functions are also susceptible to corruption. The integrity of application can benefit from identifying invalid data as early as possible.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the application perform range validation?

Procedure:

Check for unit tests.

Check thrown exceptions.

Purposely send invalid data to API(s) to test the integrity and handling of invalid data.

Example:

None.

G1340

Log all exceptional conditions.

Rationale:

Logging exceptional conditions can help to identify security problems, trace the source of the exception, and trigger security alerts.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the application perform logging of exceptional conditions?

Procedure:

Check exception handlers for logging support.

Example:

None.

G1341

Use a security manager support to restrict application access to privileged resources.

Rationale:

Desktop applications by default do not install a security manager. Installing a security manager could prevent unsecured access to resources such as the network and file system. Desktop applications can benefit from using a security manager to ensure that resources are protected.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Java Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Java Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Java Security

Evaluation Criteria:

1) Test:

Does an installed security manager restrict application access to privileged resources?

Procedure:

Check application main method for installation of a security manager.

Example:

None.

G1342

Restrict direct access to class internal variables to functions or methods of the class itself.

Rationale:

One of the primary tenets in Object Oriented Programming is encapsulation. Restricting access to internal variables not only secure the Class/Object against corruption (no data validation), it is also a maintenance issue. Hiding the implementation details allows the flexibility of underlying implementation to change.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Java Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Java Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Java Security

Evaluation Criteria:

1) Test:

Do classes directly expose internal data members?

Procedure:

Make sure all internal class variables are declared private or protected.

Example:

None.

G1343

Declare classes final to stop inheritance and prevent methods from being overridden.

Rationale:

Utility classes and classes that do not intend to be extended (classes used for user authentication) should lock down their implementation. Locking implementation can prevent methods from being overridden. Not locking down implementation can cause corruption of internal class data or allow errant code to run. For example, imagine the possibility of a class that performs credit card processing that can be overridden.

Class implementation can be locked down by declaring the class or methods final.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Java Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Java Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Java Security

Evaluation Criteria:

1) Test:

Are sensitive, security related, and utility classes declared final?

Procedure:

Check classes used in Security related processing (authentication, authorization) final keyword.

Check classes that have sensitive data (social security numbers, medical data, and salary information) for final keyword.

Check Utility classes for final keyword.

Example:

None.

G1344

Encrypt sensitive data stored in configuration or resource files.

Rationale:

Sensitive data used for application configuration files (XML), user profiles, or resource files should be protected from tampering. The sensitive data should be encrypted and or a message **digest** or checksum should be calculated to check for tampering. Application should handle generation, accessing and storing data to these files.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Application Resource Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Application Resource Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Application Resource Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Is sensitive data in configuration files and user profiles?

Procedure:

Check properties files, XML configuration files or user profiles for sensitive data in the clear.

Check for an application to edit, and creation of the file.

Example:

None.

G1346

Audit database access.

Rationale:

Auditing is critical for data access traceability. If the RDBMS was attacked, auditing is essential not only for figuring out what had occurred but also to recover lost data. Database access auditing provides logs for each access or change to the database by a given user (or an IP address for systems without user authentication).

Often current middle tier technologies (e.g., J2EE, .Net, CORBA, etc.) share database connections and may only have a single database user. Thus the burden is on the middle tier to know the identity of each user and be able to pass this information on the database (e.g., design each table to have data items such as updated by, created by, etc.).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the application database include actual user rather than database connection owner?

Procedure:

Check system documentation, database tables, and audit logs to verify that database access audit entries are created for each database access.

Example:

None

G1347

Secure remote connections to a database.

Rationale:

Just because the database is behind the corporate firewall does not mean someone inside the firewall cannot access or listen in on the wire.

Net-centricity implies that a database should be on the network and not constrained to be sitting behind an application server. This means that many unanticipated users may eventually access the database. Thus, database security should not be based on isolation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Decentralized Operations and Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is data exchanged between the database and client secure?

Procedure:

Check for secure protocol (e.g., SSL) between application and database.

Check for secure data access by IP address.

Check for configuration in the database (user) which limits user from a specified host.

Example:

None.

G1348

Log database transactions.

Rationale:

Transaction logging is generally handled by the database management system and records all changes made to the database, critical for data recovery and traceability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Are database transactions logged?

Procedure:

Commercial database management systems have a feature to log database transactions. Check to determine whether the feature has been turned on in the database management system.

Example:

None.

G1349

Validate all input that will be part of any dynamically generated SQL.

Rationale:

Not validating or filtering parameters used in dynamically generated SQL statements can lead to SQL injection attacks.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the database use filtering or data validation code?

Procedure:

Filter out character like single quote, double quote, slash, back slash, semi colon, extended character like NULL, carry return, new line, etc, in all input strings.

Example:

G1350

Implement a strong password policy for RDBMS.

Rationale:

Clean database installation often contains no passwords for root users. Also, new user accounts often defaults to no password or standard password. Having no passwords allows users access any data. Database users should always be given strong passwords. This implies a non null password, locking unused user accounts and ensuring that system user accounts are not using default passwords

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the database user table include passwords?

Procedure:

Check for null or empty values for passwords in the user table.

Use a commercially available or open source default password analysis tool to ensure that all user accounts do not retain default passwords and to ensure that all passwords are strong.

Example:

None.

G1351

Enhance database security by using multiple user accounts with constraints.

Rationale:

Constrain access to individual tables and functions by creating multiple user accounts for an application and constraining the accounts to specific functions. As a general policy, user accounts should be constrained to the minimal required database access. For example, creation of a read only account should be constrained by granting only select on the tables of interest to the read only user. This aids in password management as well as limiting the potential impact of SQL injection attacks. By granting only insert on a table, for example, and not granting select, the user could in effect create a write only database.

Each application will have different requirements in regards to grants and access to tables. If one application is compromised, it will not affect the other applications.

It also has traceability to determine which application has allowed a security violation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does each database application user have account constraints in accordance with the user function?

Procedure:

Check each database application user to ensure that the account constraints are in accordance with the user function and do not have unwarranted privileges. For example, check that read only application user accounts have only read access enabled.

Example:

None.

G1352

Use database clustering and redundant array of independent disks (RAID) for high availability of data.

Rationale:

Database clusters combined with RAID technology (e.g., data striping and mirroring) can help ensure continued operation of a system that suffers hardware or software failure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Network Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Network Infrastructure Integrity
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Availability
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the system designed to support high availability?

Procedure:

Check for the existence of a cluster and/or failover capability.

Check for the existence of RAID data storage for the database.

Example:

None.

G1356

Use the SOAP standard for all Web services.

Rationale:

The Web services security specifications are designed as an extension of SOAP. The specs are unusable without SOAP.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Web service user generate SOAP formatted XML messages?

Procedure:

Generate a test message and check it for SOAP compliance.

Example:

None.

2) Test:

Does the Web service provider generate SOAP formatted XML?

Procedure:

Generate a test message and check it for SOAP compliance.

Example:

None.

G1357

Do not rely solely on transport level security like SSL or TLS.

Rationale:

Web services inherently involve multiple intermediaries between the message sender and the ultimate destination. The intermediaries may not use transport level security. SSL and TLS do not provide end-to-end security, only security at the transport layer and only point-to-point. The use of SSL or TLS should depend on the needs of the system. For sensitive applications, augment the use of SSL/TLS with defense in depth measures such as message-level security mechanisms.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Mediate Security Assertions
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user generate encrypted XML messages?

Procedure:

Generate a test message and check it for encryption.

Example:

2) Test:

Does the Web service provider generate encrypted XML messages?

Procedure:

Generate a test message and check it for encryption.

Example:

G1359

Bind SOAP Web service security policy assertions to the service by expressing them in the associated WSDL file.

Rationale:

A Web service may be registered in zero, one, or multiple **UDDI** registries. By placing the security policy assertions in the Web service's WSDL file, they are readily available to all the consumers of the service regardless how the service was discovered

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Mediate Security Assertions
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Are Web service security policy assertions bound in the service WSDL file?

Procedure:

Check the Web Service's WSDL file for policy assertions.

Example:

None

G1362

Validate XML messages against a schema.

Rationale:

Validating messages against a schema helps prevent malicious or malformed data from compromising the integrity of a service. Validating outgoing messages against a schema helps detect compromised services. Validating messages against a schema's data attribution information also enables non-repudiation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Validate Input
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Are messages (both incoming and outgoing) validated against a schema?

Procedure:

Identify the existence of an XML Schema file and examine source code to verify that messages are checked against the schema.

Example:

None

G1363

Do not use clear text passwords.

Rationale:

Prevent a hacker from intercepting and seeing a real password.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user utilize a username/password token?

Procedure:

Generate a test message and check it for clear text passwords.

Example:

None

G1364

Hash all passwords using the combination of a timestamp, a nonce and the password for each message transmission.

Rationale:

This Guidance helps to prevent unwanted interception or discovery of clear-text-hashed passwords.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user utilize a username/password token?

Procedure:

Generate a test message and check it for a username/password token and verify that is contains a timestamp entry and a nonce entry.

Example:

None

G1365

Specify an expiration value for all security tokens.

Rationale:

Specifying an expiration value for security tokens limits the chance of being able to intercept and use a security token to impersonate an authenticated user or process.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user utilize an expiration for each security token?

Procedure:

Generate a test message and check it to make sure an expiration is associated with each security token.

Example:

None

G1366

Digitally sign all messages where non-repudiation is required.

Rationale:

Prevent hackers from changing intercepting and modifying a message.

> ***Note:*** *Non-repudiation cannot be assured with soft certificates.*

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures

Evaluation Criteria:

1) Test:

Does the Web service user digitally sign all messages?

Procedure:

Generate a test message and check it for digital signatures.

Example:

None

2) Test:

Does the Web service provider digitally sign all messages?

Procedure:

Generate a test message and check it for digital signatures.

Example:

None

G1367

Digitally sign message fragments that are required not to change during transport.

Rationale:

Signing message fragments allows the consumer of the message fragment to verify the message fragment has not changed since the producer signed the message fragment.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures

Evaluation Criteria:

1) Test:

Do message fragments sent between producers and subscribers have digital signatures when the message content must remain unchanged during transport?

Procedure:

Check system requirements for message fragments that must be transmitted unchanged between the producer and consumer. For these message fragments, check that digital signature are used to detect changes to the message fragments.

Example:

None

G1369

Digitally sign all requests made to a security token service.

Rationale:

Prevent hackers from intercepting a message and requesting a security token.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user digitally sign all messages?

Procedure:

Generate a test message and check it for digital signatures.

Example:

None

2) Test:

Does the Web service provider digitally sign all messages?

Procedure:

Generate a test message and check it for digital signatures.

Example:

None

G1371

Use the Digital Signature Standard for creating Digital Signatures.

Rationale:

Following Industry standards ensures interoperability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures

Evaluation Criteria:

1) Test:

Does the Web service user generate signatures using the Digital Signature Standard?

Procedure:

Generate a test message and check it for compliance with the Digital Signature Standard.

Example:

None

2) Test:

Does the Web service provider generate signatures using the Digital Signature Standard?

Procedure:

Generate a test message and check it for compliance with the Digital Signature Standard.

Example:

None

G1372

Use an X.509 Certificate to pass a Public Key.

Rationale:

This ensures that the owner passing the key is who he says.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service provider send a public key as part of its messages?

Procedure:

Generate a test message and check it for an X.509.

Example:

None

2) Test:

Does the Web service user send a public key as part of its messages?

Procedure:

Generate a test message and check it for an X.509.

Example:

None

G1373

Encrypt messages that cross an IA boundary.

Rationale:

Prevent hackers from reading sensitive information.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user encrypt all messages?

Procedure:

Generate a test message and check it for encryption.

Example:

None

2) Test:

Does the Web service provider encrypt all messages?

Procedure:

Generate a test message and check it for encryption.

Example:

None

G1374

Individually encrypt sensitive message fragments intended for different intermediaries.

Rationale:

Individually encrypting message fragments allows targeting individual fragments at different intermediaries along the message path to the final destination.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Are sensitive fragments of the message encrypted?

Procedure:

Observe messages that are sent to see if the sensitive fragments of the message are encrypted.

Example:

None

G1376

Do not encrypt message fragments that are required for correct SOAP processing.

Rationale:

It is possible to encrypt the entire SOAP message, various portions of the SOAP message or the contents of the data transported within the SOAP message. Encrypting the entire SOAP message requires that any intermediate processing of the SOAP message includes decryption of the entire message.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user encrypt the entire message?

Procedure:

Generate a test message and check it to make sure the XML tags are not encrypted.

Example:

None

2) Test:

Does the Web service provider encrypt the entire message?

Procedure:

Generate a test message and check it to make sure the XML tags are not encrypted.

Example:

None

G1377

Use LDAP 3.0 or later to perform all connections to LDAP repositories.

Rationale:

Using industry-proven LDAP standards help ensure interoperability of the directory repository with its consumers. LDAP v3 addresses some of the limitations of LDAP v2 in the areas of internationalization and authentication. It also allows adding new features without also requiring changes to the existing protocol through the use of using extensions and controls while maintaining backward compatibility with LDAP v2.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / LDAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / LDAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / LDAP Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Check port 636 if supporting secure LDAP (SLDAP)

Procedure:

Test the connection using an SLDAP client.

Example:

None

G1378

Encrypt communication with LDAP repositories.

Rationale:

Encryption of communication to LDAP servers helps prevent disclosure of data during transmission.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / LDAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / LDAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / LDAP Security
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Are connections to LDAP repositories encrypted?

Procedure:

Verify that connections to LDAP repository use Transport Layer Security (TLS) or Secure Sockets Layer (SSL).

Example:

G1379

Use SAML version 2.0 for representing security assertions.

Rationale:

SAML 2.0 supports **XML** assertions for supporting cross domain access and Web services. The value of this type of access is that the passing of an assertion eliminates the need to create another account in another domain.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Mediate Security Assertions
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Security Assertion Markup Language (SAML)
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Security Assertion Markup Language (SAML)
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Security Assertion Markup Language (SAML)

Evaluation Criteria:

1) Test:

Can the SAML message be validated against SAML V2.0 schema?

Procedure:

Validate SAML message against SAML V2.0.

Example:

G1380

Use the XACML 2.0 standard for SAML-based rule engines.

Rationale:

XACML-based rules can define the mechanism for creating the rule and policy set that enable meaningful **authorization** decisions. XAMCL is also integrated with **SAML** to support **role-based access control** or hierarchical resources, such as portions of XML documents.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Mediate Security Assertions
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Security Assertion Markup Language (SAML)
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Security Assertion Markup Language (SAML)
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Security Assertion Markup Language (SAML)

Evaluation Criteria:

1) Test:

Does the SAML-based rules engine use the XACML 2.0 standard?

Procedure:

Emulate a rule and run against rule engine using SOAP messaging.

Example:

G1381

Encrypt sensitive persistent data.

Rationale:

When data is persisted, there is always a chance that the security of the system that stores the data may be compromised. To minimize the risk, all sensitive data such as passwords and personal information should be encrypted when it is persisted.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Cryptography
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Cryptography
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Data at Rest

Evaluation Criteria:

1) Test:

Is all sensitive data that is persisted encrypted?

Procedure:

Look at all data stores and check for encrypted passwords and other sensitive data..

Example:

G1382

Be associated with one or more Communities of Interest (COIs).

Rationale:

The DoD Net-Centric Data Strategy emphasizes the establishment of Communities of Interest (**COI**s). This strategy introduces management of data within Communities of Interest (COIs) rather than standardizing **data elements** across the DoD. Thus all DoD Programs must map to one of more COIs. DoD Programs should participate in COIs as a normal course of doing business. They will identity relevant COIs; actively collaborate with them to promote reuse and cross-coordination of **metadata**; sponsor participation of system developers in the COI process and where appropriate contribute engineering expertise to the COI as a stakeholder. New programs should include community collaboration requirements in acquisition documents as required.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Is the Program associated with a **COI**?

Procedure:

Check the DoD Metadata registry to determine whether program is associated with any **COI**(s).

Example:

None

G1383

Use a registered namespace in the XML Gallery in the DoD Metadata Registry.

Rationale:

The use of the **DoD Metadata Registry** helps to avoid name collisions and conflicts.

The assignation of a unique **registered namespace** permits a program to be uniquely identified and categorized. The DoD **Net-Centric Data Strategy** requires that data products be stored in shared spaces to provide access to all authorized users and that these data products be tagged with **metadata** to enable discovery of data by authorized users. The use of a unique registered namespace provides an absolute identifier to products associated with a particular product and is an **XSD** schema requirement.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry

Evaluation Criteria:

1) Test:

Does the Program have an assigned namespace for its XML data assets?

Procedure:

Check **DoD Metadata Registry** to determine whether the Program is associated with **COI**(s).

Example:

None

G1384

Review XML Information Resources in the DoD Metadata Registry, using those which can be reused.

Rationale:

The DoD Net-Centric Data Strategy requires that **XML** information resources within a **COI** in the **DoD Metadata Registry** be examined by DoD projects for possible reuse to help foster common standards within a **COI** and promote interoperability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling

NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Has the program reused information resources from the **DoD Metadata Registry**?

Procedure:

Check the **XSDs** associated with the program to determine whether XSDs referenced by other namespaces have been used. Check the **DoD Metadata Registry** to determine whether the Program has registered the reuse of XML information resources belonging to other namespaces. Reuse is indicated by formally subscribing to selected components in the registry.

Example:

None

G1385

Identify XML Information Resources for registration in the XML Gallery of the DoD Metadata Registry.

Rationale:

The DoD Net-Centric Data Strategy requires that **XML Information Resources** developed during the course of a program be identified, examined for usefulness by other DoD Programs in the same or related **COIs** and be submitted for inclusion in the XML Gallery of the **DoD Metadata Registry**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry

Evaluation Criteria:

1) Test:

Has the Program submitted new information resources to the **DoD Metadata Registry**?

Procedure:

Check the **XSDs** associated with the program namespace to determine whether they have been registered in the **DoD Metadata Registry** XML Gallery.

Example:

None

G1386

Review predefined commonly used data elements in the Data Element Gallery of the DoD Metadata Registry, using those in the relational database technology which can be reused in the Program.

Rationale:

The DoD Net-Centric Data Strategy requires that DoD Programs examine data element information resources within a **COI** in the **DoD Metadata Registry** for possible reuse to help foster common standards within a **COI** and promote interoperability. Elements include **`US State Codes`** and **`Country Codes`**. This reuse is preferential to reusing existing industry standard **data elements** or developing new **data elements**.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Has the Program reused common database elements?

Procedure:

Check the DoD Metadata Registry Data Element Gallery to determine whether the program has registered database elements for reuse. Reuse is indicated by formally subscribing to selected components in the registry.

Check the program database to see whether registered have been included therein.

Example:

None

G1387

Identify data elements created during Program development for registering in the Data Element Gallery of the DoD Metadata Registry.

Rationale:

The DoD Net-Centric Data Strategy requires that Programs identify and examine developed **data elements** for usefulness by other DoD Programs in the same or related **COIs** and submit the data elements for inclusion in the **Data Element Gallery** of the **DoD Metadata Registry**.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry

Evaluation Criteria:

1) Test:

Has the Program submitted common database elements to the **DoD Metadata Registry**?

Procedure:

Check the DoD Metadata Registry Data Element Gallery to determine whether the program has submitted database elements for reuse.

Example:

None

G1388

Use predefined commonly used database tables in the DoD Metadata Registry.

Rationale:

The DoD Net-Centric Data Strategy requires that DoD Programs examine data table information resources within a **COI** in the **DoD Metadata Registry** for possible reuse to help foster common standards within a COI and promote interoperability. This reuse is preferable to reusing existing industry standard **data elements** or developing new data elements. Some examples are **`Country Code`**, **`US State Code`**, **`Purchase Order Type Code`**, **`Security Classification Code`**. These tables are found in the **Reference Data Set** Gallery of the DoD Metadata Registry.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Has the Program reused common database tables?

Procedure:

Check the DoD Metadata Registry to determine whether the program has registered database tables for reuse. Reuse is indicated by formally subscribing to selected components in the registry.

Check the program database to see whether registered data tables have been included therein.

Example:

None

G1389

Publish database tables which are of common interest by registering them in the Reference Data Set Gallery of the DoD Metadata Registry.

Rationale:

The DoD Net-Centric Data Strategy requires that DoD Programs identify and examine developed data tables for usefulness by other DoD Programs in the same or related **COIs** and be submit the data elements for inclusion in the **Reference Data Set** Gallery of the **DoD Metadata Registry**.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry

Evaluation Criteria:

1) Test:

Has the Program submitted common database tables to the DoD Metadata Registry?

Procedure:

Check the DoD Metadata Registry Reference Data Set Gallery to determine whether the program has submitted database tables for reuse.

Example:

None

G1390

Standardize on the terminology published by relevant Communities of Interest (COIs) listed in the Taxonomy Gallery of the DoD Metadata Registry.

Rationale:

A **taxonomy** partitions the body of knowledge associated with a **Community of Interest COI** and defines the relationships among component parts. A taxonomy permits classification of concepts associated with a COI. This in turn provides categories and definitions for discovery tags which aids in information use and retrieval by authorized users. Program use of COI taxonomies occurs in several places:

1. Taxonomy used to describe information services for discovery.
2. Taxonomies created by the COI as a means to extend the **DoD Discovery Metadata Specification** (**DDMS**) for data asset discovery.
3. Taxonomies used to support mediation.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Has the Program adhered to the standard **taxonomies** for the **COIs** associated with the program?

Procedure:

Check the DoD Metadata Registry and Taxonomy Gallery to determine whether taxonomies exist for the COI in which the Program resides.

Example:

None

G1391

Identify taxonomy additions or changes in conjunction with the Communities of Interest (COIs) during the Program development for potential inclusion in the Taxonomy Gallery of the DoD Metadata Registry.

Rationale:

DoD Programs associated with a specific COI need to identify and submit potential taxonomy changes or additions to the **DoD Metadata Registry** to maintain an accurate and effective taxonomy within the **COI**.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Has the Program submitted **taxonomy** additions or changes to the **DoD Metadata Registry**?

Procedure:

Check the DoD Metadata Registry and to determine whether the program has submitted taxonomy changes for reuse.

Example:

None

G1566

Use `alt` attributes to provide alternate text for non-text items such as images.

Rationale:

This usage aids users in understanding the Web page even if their browsers cannot display images.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Are alt attributes provided for non-text content?

Procedure:

Check for the existence of alt attributes for all Web site non-text content.

Example:

None.

G1569

Maintain a comprehensive list of all of the Components that are part of the Node.

Rationale:

Throughout the lifecycle of a Node (from design to instantiation), this action is fundamental to the provisioning of a shared infrastructure and the avoidance of functional duplication within the Node. This activity has a direct impact on the design and implementation requirements during acquisition.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Nodes as Stakeholders
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Is there a list of Components that comprise the Node?

Procedure:

Examine the documents (for example, the Node's design requirements) and look for a list of Components.

Example:

None.

G1570

Assume an active management role among the Components within the Node.

Rationale:

Involvement of the Node as a stakeholder in its Components (from design to instantiation) has a bearing on **Global Information Grid** (GIG) interoperability. Strong coordination among a Node's Components will likely avoid the external exposure of inconsistencies or, worse, incomplete, inaccurate, or misunderstood data.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Nodes as Stakeholders
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Do the Components of the Node set forth requirements in their [appropriate acquisition document] for coordinating with the Node.

Procedure:

Check the [appropriate acquisition document] of the Components and determine if the Node is listed as a stakeholder or if there are requirements for coordinating with the Node.

Example:

A Component's **Capability Development Document** (**CDD**) may state a requirement for participating in a Node which could satisfy this requirement.

2) Test:

Do the Components of the Node list the Node as a primary stakeholder in their [appropriate acquisition document]?

Procedure:

Check the [appropriate acquisition document] of the Components and determine if the Node is listed as a stakeholder or if there are requirements for coordinating with the Node.

Example:

A Component's **Capability Development Document** (**CDD**) may state a requirement for participating in a Node which could satisfy this requirement.

G1571

Maintain a comprehensive list of all the Communities of Interest (COIs) to which the Components of a Node belong.

Rationale:

The Node infrastructure must be engineered to support the information exchange between **Communities of Interests** (**COIs**). If a comprehensive list of COIs is not created and maintained then the infrastructure may no longer be adequate and may continue to make provisions for COIs that are no longer a part of the Node.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs

Evaluation Criteria:

1) Test:

Do the Node's Components have representation registered within the DoD Metadata Registry as members of the Communities of Interest (COIs)?

Procedure:

Examine the DoD Metadata Registry for members of the Node organization that are members of the pertinent COIs.

Example:

None.

G1572

Include the Node as a party to any Service Level Agreements (SLAs) signed by any of the components of the Node.

Rationale:

The Node has a stake in performance specifications provided in the **Service Level Agreements** (**SLA**). Since the SLA is a contract that commits the application service provider to a required level of service. The Node must be able to support that level of service with its infrastructure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Availability

Evaluation Criteria:

1) Test:

Does the Node have copies of all Service Level Agreements (SLAs) signed by its Components?

Procedure:

Compare the Service Level Agreements (SLAs) against the service Components supported by the Node.

Example:

None.

G1573

Define the enterprise design patterns that a Node supports.

Rationale:

The Node infrastructure must be engineered to support information exchanges between various **Communities of Interest** (COIs). The COIs can require any number of **Components** to fulfill the COIs mission, When a Component wishes to make its data available over the **enterprise**, there are different enterprise design pattern which can be used. For example, the mechanism selected by a Component to exchange information may be publish-subscribe, broker, or client server. The Node infrastructure must support whichever enterprise design pattern mechanism is selected.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Node document which types of enterprise design patterns it supports?

Procedure:

Look through the Node documents for a list of enterprise design patterns it supports.

Example:

None.

G1574

Define which enterprise design patterns a Component requires.

Rationale:

A Component should document which enterprise design patterns it intends to capitalize on to meet its mission. For example, a client interested in using a client-server weather service, could have problems if the weather service is a real-time publish-subscribe service. This action clarifies for the Node which enterprise design patterns are required by its Components and provides direction for which patterns to support at the Node level.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Component indicate which type of enterprise design pattern it will use?

Procedure:

Look through the Component documentation and that defines what type of enterprise design pattern it uses.

Example:

None.

G1575

Designate Node representatives to relevant Communities of Interest (COIs) in which Components of the Node participate.

Rationale:

COI is the inclusive term used to describe collaborative groups of users who must exchange information in pursuit of their shared goals, interests, missions, or business processes and who therefore must have shared vocabulary for the information they exchange. The principal mechanism for recording COI agreements is the **DoD Metadata Registry** required by the DoD CIO Memorandum *DoD Net-Centric Data Management Strategy: Metadata Registration*. There are registry implementations on the Non-secure Internet Protocol Router Network (**NIPRNet**), Secret Internet Protocol Router Network (**SIPRNet**), and Joint Worldwide Intelligence Communications System (**JWICS**).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs

Evaluation Criteria:

1) Test:

Does the Node have representation registered within the Metadata Registry as members of the **Communities of Interest** (**COIs**)?

Procedure:

Examine the **DoD Metadata Registry** for members of the Node organization that are members of the pertinent COIs.

Example:

None.

G1576

Provide an environment to support the development, build, integration, and test of net-centric capabilities.

Rationale:

Nodes should provide an environment to support the development, integration, and testing of net-centric capabilities of its **Components**. As Nodes themselves and the Components within the Nodes move closer to the implementation of net-centric capabilities, it becomes increasingly important to provide a development, integration, and test environment to support those capabilities. This environment should allow for the exercise not just the Node infrastructure, but also either host locally within the Node, or provide access to, **Net-Centric Enterprise Services** (NCES) piloted services. The particulars on how this is done depend on the characteristics of the Node. For example, mobile or deployed Nodes would provide environments substantially different than fixed land-based or permanent Nodes.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Internal Component Environment
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

Evaluation Criteria:

1) Test:

Are there instructions on how to develop, build, integrate or test Components within the Node?

Procedure:

Look for user guides or installation instructions that cover the Node environment.

Example:

None.

G1577

Maintain an Enterprise Service schedule for interim and final enterprise capabilities within the Node.

Rationale:

The current state of **Enterprise Services** is in flux. Developing **Components** that rely on those services can create a circular problem for development. An enterprise service schedule for interim and final capabilities will help elevate the co-dependencies of the Component lifecycle from the Node lifecycle.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Internal Component Environment
NESI / Part 4: Node Guidance / General Responsibilities / Coordination of Node and Enterprise Services
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

Evaluation Criteria:

1) Test:

Is there an enterprise service schedule or roadmap that covers interim and final capabilities of the Node?

Procedure:

Look for the existence of the schedule or a roadmap for the Node.

Example:

None.

G1578

Define a schedule for Components that includes the use of the Enterprise Services defined within the Node's enterprise service schedule.

Rationale:

The exercise of matching those **Enterprise Services** required by the **Component** to those provided by the Node can help identify and gaps in the Node's functionality. By tying the Component's enterprise services to the Node's **enterprise** schedule, critical paths may be identified in the Node's schedule.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Internal Component Environment
NESI / Part 4: Node Guidance / General Responsibilities / Coordination of Node and Enterprise Services
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

Evaluation Criteria:

1) Test:

Does the Component have an enterprise service schedule or roadmap that shows the progression of enterprise service usage by interim and final capabilities of the Component?

Procedure:

Look for the existence of the schedule or a roadmap for the Component.

Example:

None.

G1579

Define which Enterprise Services the Node will host locally when the Node becomes operational.

Rationale:

Locally defined **Enterprise Services** are inherently faster and less susceptible to network failures and traffic than local services. If a **Component** requires performance based or critical enterprise services that the Node will only provide as a **proxy**, then development, building, integration and testing should be done to the local enterprise service specification. If the Node developed enterprise service will not be ready until near the end of the Component's schedule, take steps to minimize risk.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Internal Component Environment
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Node specification identify which Enterprise Services will be locally defined within the Node?

Procedure:

Review the Node specification for a list of Enterprise Services that will be locally defined within the Node.

Example:

None.

G1580

Define which Enterprise Services will be hosted over the Global Information Grid (GIG) when the Node becomes operational.

Rationale:

Enterprise Services that are defined using **proxies** should have interfaces that follow the standards defined by the enterprise service provider. Therefore, the access to the **server** should be fairly stable and almost static in nature with few changes. These are services that should be in the critical path of a Component's mission.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Internal Component Environment
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Node specification identify which Enterprise Services will be defined using proxies?

Procedure:

Review the Node specification for a list of Enterprise Services that will be defined using proxies.

Example:

None.

G1581

Expose legacy functionality through the use of a service.

Rationale:

Nodes might contain **legacy systems** or **applications** that are in the **Sustainment** lifecycle phase. These **components** are often referred to as ***legacy*** systems or applications. If a Node needs to expose functionality or data from the legacy component, changing the internals of such components to support net-centricity is often impractical with little return on investment. In these cases, it is often desirable to offer a reasonable interim solution by exposing the functionality through the use of well known patterns (such as a **facade design pattern**).

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Integration of Legacy Systems
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the Node use **facade design patterns** such as the wrapper or adapter pattern to expose the functionality of legacy systems or applications?

Procedure:

Make sure that all the Components that are exposed to the internal Node Components or to the external network (with the Node as a proxy) use a facade design pattern such as wrapper or adapter.

Example:

None.

G1582

In Node Enterprise Service schedules, include version numbers of standard Enterprise Services interfaces being implemented.

Rationale:

Given the complexity, varied implementation timing, and leading edge nature of **Enterprise Services**, the **orchestration** of efforts is essential for the successful integration of the Node's Components. The dependencies captured by such a schedule should clearly show what capabilities will be available and when during the Node's lifecycle.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Coordination of Node and Enterprise Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Are Enterprise Services interface versions provided on the enterprise service schedule for the Node?

Procedure:

Review the Enterprise Services schedule published for the Node and make sure the schedule provides necessary details including specific version numbers, workarounds, assumptions, constraints and configuration limitations that are interwoven into the schedule.

Example:

An Enterprise Service might be releasing a new version during the lifecycle of the Node's development; which version's functionality will be available when is essential for the successful integration of the Node's Components.

2) Test:

Are Enterprise Services interface versions provided on the enterprise service schedule for the Component?

Procedure:

Review the Enterprise Services schedule published for the Component and make sure the schedule provides necessary details including specific version numbers, workarounds, assumptions, constraints and configuration limitations that are interwoven into the schedule.

Example:

An Enterprise Service might be releasing a new version during the lifecycle of the Node's development; which version's functionality will be available when is essential so the Component can utilize the appropriate available capabilities.

G1583

Provide routine Enterprise Services schedule updates to every component of a Node.

Rationale:

A fundamental justification for the existence of nodes is to ensure it provides a shared infrastructure for its components. If that infrastructure evolves independently of the components, then they may be developed at timeframes and rates of evolution that differ from the capabilities of the available shared infrastructure. In addition, components may be members of multiple Nodes, providing an additional coordination challenge. Regular updates to the components of the master schedule will assist in managing this challenge.

Referenced By:

NESI / Part 4: Node Guidance / General Responsibilities / Coordination of Internal Components
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Are there multiple iterations of the Enterprise Services schedule developed over time and is the most recent update timely?

Procedure:

Check for version numbering and release dates of the Enterprise Services schedule. Ensure that a reasonably recent update is available.

Example:

None.

G1584

Provide a transport infrastructure that is shared among components within the Node.

Rationale:

Transport elements provided by the Node are a means for the Node to implement **Global Information Grid** (**GIG**) **Information Assurance** (**IA**) boundary protections, bind Components together, and satisfy other enterprise requirements. As transport elements are an essential piece of the net-centric puzzle, they also play a key role in minimizing interoperability issues. A Node's provisioning of the shared transport and related guidance is a key aspect of its existence.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport
NESI / Part 4: Node Guidance / Node Transport
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal

Evaluation Criteria:

1) Test:

Does the Node's design provide for a transport infrastructure?

Procedure:

Review the Node's infrastructure design and ensure that the Node provides the necessary transport elements for shared use by its Components.

Example:

None.

2) Test:

Are the Node's Components using the Node provisioned transport infrastructure?

Procedure:

Review the design of the Node's Components (see G1569) and ensure that they all utilize the common transport infrastructure of inter-Nodal communication.

Example:

None.

G1585

Provide a transport infrastructure for the Node that implements Global Information Grid (GIG) Information Assurance (IA) boundary protections.

Rationale:

The **Global Information Grid** (**GIG**) is intended to be the ***outside world*** for all the components within the Node. In order to protect the components within the Node from the outside world and to protect the outside world from the Node, the Node should control the **IA** Boundary.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport
NESI / Part 4: Node Guidance / Node Transport
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Net-Centric IA Posture and Continuity of Operations
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal

Evaluation Criteria:

1) Test:

Is there an IA device in the acquisition list?

Procedure:

Look for an IA device within the parts list for the Node.

Example:

None.

2) Test:

Is the IA device configured to meet security requirements?

Procedure:

Check the Node's IA installation guide and look for procedures that describe how to configure the IA device for the Nodes particular needs.

Example:

None.

G1586

Provide a transport infrastructure for the Node that is Internet Protocol Version 6 (IPv6) capable in accordance with the appropriate governing transition plan.

Rationale:

During the transition period in the DoD community (FY06-FY15) networks, services and applications will be in a mixed environment. All Critical **Key Performance Parameters** (**KPPs**) must be able to operate in an **Internet Protocol Version 4** (**IPv4**) only network, an **Internet Protocol Version 6** (**IPv6**) only network, and a dual-stack network.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the Node provide a transport infrastructure that is Internet Protocol Version 6 (IPv6) capable?

Procedure:

Verify that the Node transport infrastructure supports IPv6 such that Node Components are able to complete all critical functions utilizing only IPv6 on the network (with no use of IPv6 over IPv4 tunneling).

Example:

None.

G1587

Prepare an Internet Protocol Version 6 (IPv6) transition plan for the Node.

Rationale:

The transition from **Internet Protocol Version 4** (**IPv4**) to **Internet Protocol Version 6** (**IPv6**) is non-trivial and requires a great deal of coordination and effort on the part of everyone involved. The transition plan helps to minimize the potential disastrous side effects of the transition.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Is there an Internet Protocol Version 6 (IPv6) transition plan for the Node?

Procedure:

Look for an Internet Protocol Version 6 (IPv6) transition plan document.

Example:

None.

G1588

Coordinate an Internet Protocol Version 6 (IPv6) transition plan for a Node with the Components that comprise the Node.

Rationale:

The effects of the transition from **Internet Protocol Version 4** (**IPv4**) to **Internet Protocol Version 6** (**IPv6**) is isolated in the Node infrastructure but can have impacts on all the **Components** that comprise the Node. The transition Plan should cover a "window" that allows all the Components to operate in either IPv4 or IPv6 (i.e., **Dual Stack Mode**) to make the transition.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the plan allow for a **Dual Stack** environment at least during some transition period?

Procedure:

Look for a part of the transition plan that addresses **Dual Stack** mode of operation.

Example:

None.

G1589

Address issues in the appropriate governing Internet Protocol Version 6 (IPv6) transition plan as part of the IPv6 Transition Plan for a Node.

Rationale:

DoD has mandated that each service create an **IPv6** transformation office to manage the transition to IPv6. Node transition plans must be aligned and in conformance with the appropriate governing office's plans or criteria.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the Node's IPv6 Transition Plan have a section that addresses specific criteria established by the appropriate governing IPv6 transition office or plan?

Procedure:

Review the IPv6 plan for a section or specific criteria that address the appropriate items from the appropriate governing plan or is approved by the appropriate governing office.

Example:

The Air Force IPv6 Transition Office requires each program to develop a plan with approval by the transition office (in lieu of aligning with a central plan). To check an Air Force Node's alignment, look to see that the Node's IPv6 transition plan is approved by the appropriate authority.

G1590

Include transition of all the impacted elements of the network as part of the Internet Protocol Version 6 (IPv6) Transition Plan for a Node.

Rationale:

Internet Protocol Version 6 (**IPv6**) transition has an impact on many transport infrastructure **Components**. The Node's IPv6 Transition Plan should include transition of all impacted network elements including **DNS**, routing, security, and dynamic address assignment. The *DoD IPv6 Network Engineer's Guidebook* (Draft) and the *DoD IPv6 Application Engineer's Guidebook* (Draft) provide guidance for transition of impacted Components.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the Internet Protocol Version 6 (IPv6) Transition Plan address the impact of the transition to IPv6 on the Domain Name Service (DNS)?

Procedure:

Review the plan and look for a section dedicated to the Domain Name Service (DNS). At a minimum, it should indicate that there is no impact.

Example:

None.

2) Test:

Does the Internet Protocol Version 6 (IPv6) Transition Plan address the impact of the transition to IPv6 on routing?

Procedure:

Review the plan and look for a section dedicated to routing. At a minimum, it should indicate that there is no impact.

Example:

None.

3) Test:

Does the Internet Protocol Version 6 (IPv6) Transition Plan address the impact of the transition to IPv6 on security?

Procedure:

Review the plan and look for a section dedicated to security. At a minimum, it should indicate that there is no impact.

Example:

None.

4) Test:

Does the Internet Protocol Version 6 (IPv6) Transition Plan address the impact of the transition to IPv6 on dynamic address assignment?

Procedure:

Review the plan and look for a section dedicated to dynamic address assignment. At a minimum, it should indicate that there is no impact.

Example:

None.

G1591

Prepare IPv6 Working Group products as part of the Internet Protocol Version 6 (IPv6) transition plan for a Node.

Rationale:

The **Internet Protocol Version 6** (**IPv6**) Working Group has prescribed various products that can aid in the planning for the transition from **Internet Protocol Version 4** (**IPv4**) to IPv6. The Node's Transition Plan should prepare these products to ensure that all the required activities are addressed.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Are the Internet Protocol Version 6 (IPv6) Working Group products in the Node's Transition Plan?

Procedure:

Look for the Working Group products in the Node's Transition Plan.

Example:

None.

G1592

Include interoperability testing in the plan as part of the Internet Protocol Version 6 (IPv6) transition plan for a Node.

Rationale:

During the **DoD** transition period, a mixed **IPv4/IPv6** environment will exist. Interoperability testing with both standards will ensure the Node can fully function during the transition period with all other Nodes.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the Node's IPv6 transition plan address interoperability testing in a mixed environment?

Procedure:

Review the transition plan and verify that a test plan exists that specifically addresses interoperability testing in a mixed IP environment.

Example:

None.

G1595

Implement Domain Name System (DNS) to manage hostname/address resolution within the Node.

Rationale:

Using **Domain Name System** (**DNS**) obviates the need for hard-coding **Internet Protocol** (**IP**) addresses within the Node. In addition, DNS servers local to the Node allow for stable access of replicated entries from outside the Node.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Are there any hard coded Internet Protocol (IP) addresses within the source code or data files?

Procedure:

Look at the source code, properties files and descriptor files for the occurrence of Internet Protocol Version 4 (IPv4) or Internet Protocol Version 6 (IPv6) Internet Protocol (IP) addresses.

Example:

None.

2) Test:

Is there a Domain Name System (DNS) server in the Node acquisition list?

Procedure:

Look for a Domain Name System (DNS) server within the parts list for the Node.

Example:

None.

G1596

Use Domain Name System (DNS) Mail eXchange (MX) Record capabilities to configure electronic mail delivery to the Node.

Rationale:

Utilizing the **Domain Name System** (**DNS**) **Mail eXchange (MX) record** capability will avoid the need to hard code delivery routes and instructions within a Node's email system and buffers it from physical changes made to email delivery points and routes outside of the Node. The DNS MX record is a standard and commonly accepted mechanism for resolving email delivery routes and addresses across the Internet.
Internet Engineering Task Force (**IETF**) Request for Comments (RFC) 2821 of April 2001 established rules for MX record usage.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Are there **Mail eXchange (MX) Records** defined within the **Domain Name System** (**DNS**)?

Procedure:

Look at the Domain Name System (DNS) records for Mail eXchange (MX) Records.

Example:

None.

G1598

Allow dynamic Domain Name System (DNS) updates to the Node's internal DNS service by local Dynamic Host Configuration Protocol (DHCP) server(s).

Rationale:

There are two basic methods for assigning of **Internet Protocol** (**IP**) addresses within a network: static and dynamic. Static addresses are assigned to a particular system and never change. Dynamic Internet Protocol (IP) addresses are issued for a variable length of time: the ***DCHP lease time***. **Dynamic Host Configuration Protocol** (**DHCP**) is the principle mechanism used to assign and manage dynamic IP addresses. If the DHCP servers are allowed to update the **Domain Name System** (**DNS**), then the number of static addresses required by the system can be drastically reduced with preference being given to requesting services by domain name rather than IP address.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Dynamic Host Configuration Protocol (DHCP)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Dynamic Host Configuration Protocol (DHCP)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the Domain Name System (DNS) server in the Node acquisition list support updates from Dynamic Host Configuration Protocol (DHCP) Servers?

Procedure:

Review the Domain Name System (DNS) server specification to confirm that it supports such operations.

Example:

None.

G1599

Simultaneously support Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) in the Node's Domain Name System (DNS) service.

Rationale:

During the transition period in the DoD community (FY06-FY15) networks, services and applications will be in a mixed environment. The Domain Name System (DNS) returns different address records depending on the Internet Protocol (IP) environment: A records for IPv4 or AAAA records for IPv6. A DNS must be able to support both.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the Domain Name System (DNS) server support both A and AAAA records?

Procedure:

Review the Domain Name System (DNS) specification to confirm that it supports both A and AAAA records.

Example:

None.

G1600

Obtain Internet Protocol Version 6 (IPv6) addresses to use for DoD IP addressable resources from DISA.

Rationale:

All the **Internet Protocol** (**IP**) addresses in use on a DoD network must be from an appropriate clearing house in order to maintain control and accountability on the network. **DISA** is the clearing house for all DoD addresses.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP)
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP)
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Is there a proper entry in the Military Network Information Center (MILNIC) for every IP address assigned to the system?

Procedure:

Verify an adequate address allocation has been made in the Military Network Information Center (MILNIC) for the system.

Example:

None.

G1601

Use configurable routers to provide dynamic Internet Protocol (IP) address management using the Dynamic Host Configuration Protocol (DHCP).

Rationale:

There are two basic methods for assigning of **Internet Protocol** (**IP**) addresses within a network: static and dynamic. Static addresses are assigned to a particular system and never change. Dynamic IP addresses are issued for a variable length of time: the ***DCHP lease time***. The **Dynamic Host Configuration Protocol** (**DHCP**) is the principle mechanism used to assign and manage dynamic IP addresses.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Subnets and Overlay Networks / Broadcast, Multicast, and Anycast
NESI / Part 4: Node Guidance / Node Transport / Subnets and Overlay Networks / Broadcast, Multicast, and Anycast
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Dynamic Host Configuration Protocol (DHCP)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Dynamic Host Configuration Protocol (DHCP)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the router in the Node acquisition list support Dynamic Host Configuration Protocol (DHCP)?

Procedure:

Review the router specification to confirm that it supports such operations.

Example:

None.

G1602

Use configurable routers to provide static Internet Protocol (IP) addresses.

Rationale:

Some network **Components** such as the **routers** themselves and other security related services must reside on static **Internet Protocol** (**IP**) addresses. Serious comprises in the network can arise if these services are allowed to be dynamic.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the **router** in the Node acquisition list support static **Internet Protocol** (**IP**) addressing?

Procedure:

Review the router specification to confirm that it supports such operations.

Example:

None.

G1604

Use configurable routers to provide time synchronization services using Network Time Protocol (NTP).

Rationale:

Over time, most computer clocks drift. **Network Time Protocol** (**NTP**) is one way to ensure that a computer clock stays accurate. Unfortunately, in order to stay synchronized, a network connection needs to be maintained. In environments that have limited bandwidth or poor **quality of service** (**QoS**) this can become a major issue.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Network Time Service
NESI / Part 4: Node Guidance / Node Transport / Network Services / Network Time Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the **router** in the Node acquisition list support NTP Service?

Procedure:

Review the routers specification to confirm that it supports such operations.

Example:

None.

G1605

Use configurable routers to provide multicast addressing.

Rationale:

Multicast addresses identify interfaces that allow a packet to be sent to all the addresses registered for the multicast service. This allows network to easily support applications such as **collaboration**, audio and video.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the **router** in the Node acquisition list support NTP Service?

Procedure:

Review the router specification to confirm that it supports such operations.

Example:

None.

G1606

Manage routers remotely from within the Node.

Rationale:

Router manufactures routinely provide tools to enable remote, over the network, router configuration and management in addition to a local console within the **Node**. These tools can speed and centralize the administration of the routers in a Node.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Decentralized Operations and Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the **router** in the Node acquisition list support remote management?

Procedure:

Review the router specification to confirm that it supports such operations.

Example:

None.

G1607

Configure routers according to National Security Agency (NSA) Router Security Configuration guidance.

Rationale:

The *Router Security Configuration Guide* provides technical guidance intended to help network administrators and security officers improve the security of their networks. It contains principles and guidance for secure configuration of **Internet Protocol** (**IP**) routers, with detailed instructions for Cisco System routers. The information presented can be used to control access, help resist attacks, shield other network **Components**, and help protect the integrity and confidentiality of network traffic.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Concurrent Transport of Information Flows
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Is the **Router** Security Checklist complete and up to date?

Procedure:

Check for the occurrence of the checklist; there should be a copy for every time the checklist has been completed. The checklist should indicate the date, time and results of the checklist with recommendation actions.

Example:

Router Security Checklist
This security checklist is designed to help review router security configuration and remind a user of any security areas that might be missed.

- Router security policy written, approved, distributed.
- Router IOS version checked and up to date.
- Router configuration kept off-line, backed up, access to it limited.
- Router configuration is well-documented, commented.
- Router users and passwords configured and maintained.
- Password encryption in use, enable secret in use.
- Enable secret difficult to guess, knowledge of it strictly limited. (if not, change the enable secret immediately)
- Access restrictions imposed on Console, Aux, VTYs.
- Unneeded network servers and facilities disabled.
- Necessary network services configured correctly (e.g. DNS)
- Unused interfaces and VTYs shut down or disabled.
- Risky interface services disabled.

Part 2: Traceability

- Port and protocol needs of the network identified and checked.
- Access lists limit traffic to identified ports and protocols.
- Access lists block reserved and inappropriate addresses.
- Static routes configured where necessary.
- Routing protocols configured to use integrity mechanisms.
- Logging enabled and log recipient hosts identified and configured.
- Router's time of day set accurately, maintained with NTP.
- Logging set to include consistent time information.
- Logs checked, reviewed, archived in accordance with local policy.
- SNMP disabled or enabled with good community strings and ACLs.

G1608

Obtain reference time from a standard globally synchronized time source.

Rationale:

Currently, Network Time Service is not a homogeneous service across the **Global Information Grid** (**GIG**). Security directives prevent **IP**-based time synchronization across **firewall** boundaries (e.g., AFI 33-115, 16). An example of a precise globally synchronized time source is a **Global Positioning System** (**GPS**) system.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Network Time Service
NESI / Part 4: Node Guidance / Node Transport / Network Services / Network Time Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the acquisition list include a precise globally synchronized time source such as a **Global Positioning System** (**GPS**)?

Procedure:

Review the acquisition list for a precise globally synchronized time source such as a **Global Positioning System** (**GPS**) that can provide accurately synchronized time.

Example:

None.

G1609

Arrange for a backup time source.

Rationale:

Use one or more backup time sources. The most common type of backup time sources are crystal oscillators. The physical characteristics of the piezoelectric quartz crystal produce electrical oscillations at an extremely accurate frequency which can be used to mark time.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Network Time Service
NESI / Part 4: Node Guidance / Node Transport / Network Services / Network Time Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the acquisition list include a backup time source?

Procedure:

Review the acquisition list for a backup time system that can be used to synchronize time accurately.

Example:

Crystal oscillator examples include cesium or rubidium. The following table shows crystal oscillator types:

MCXO	microcomputer-compensated crystal oscillator
OCVCXO	oven-controlled voltage-controlled crystal oscillator
OCXO	oven-controlled crystal oscillator
RbXO	rubidium crystal oscillators (RbXO)
TCVCXO	temperature-compensated-voltage controlled crystal oscillator
TCXO	temperature-compensated crystal oscillator
VCXO	voltage-controlled crystal oscillator

G1610

Configure the Dynamic Host Configuration Protocol (DHCP) services to assign multicast addresses.

Rationale:

When **Dynamic Host Configuration Protocol** (**DHCP**) services assign temporary **Internet Protocol** (**IP**) addresses to clients, the clients may wish to participate in a **multicast** service. Therefore, the DHCP service must support the assignment of multicast addresses as part of normal operations.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Subnets and Overlay Networks / Broadcast, Multicast, and Anycast
NESI / Part 4: Node Guidance / Node Transport / Subnets and Overlay Networks / Broadcast, Multicast, and Anycast
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Dynamic Host Configuration Protocol (DHCP)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Dynamic Host Configuration Protocol (DHCP)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the **router** in the Node acquisition list support the assignment of **multicast** Internet Protocol (**IP**) addresses as part of the normal **Dynamic Host Configuration Protocol** (DHCP) service?

Procedure:

Review the **router** specification to confirm that it supports such operations.

Example:

None.

G1611

Implement Internet Protocol (IP) gateways to interoperate with the Global Information Grid (GIG) until IP is supported natively for Components that are not IP networked.

Rationale:

Component systems such as aircraft data links (**Link-16**, **SADL**, etc), should implement **Transmission Control Protocol/Internet Protocol** (**TCP/IP**) gateways to interoperate with the **Global Information Grid** (**GIG**) until TCP/IP is supported natively. This acts as an interim step that can be used to bridge the **Internet Protocol** (**IP**) divide.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Integration of Non-IP Transports
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Integration of Non-IP Transports
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Are **Internet Protocol** (**IP**) and non-IP networks connected via gateways?

Procedure:

Verify IP and non-IP networks are connected via one or more gateways.

Example:

1. Identify gateways between IP and non-IP networks within DoDAF diagrams.

2. Verify successful data translation between IP and non-IP networks via a gateway such as verifying track data transmission between a Link 16 equipped user and a GIG edge IP router.

G1613

Prepare a Node to host new Component services developed by other Nodes or by the enterprise itself.

Rationale:

A key aspect of an open systems approach to interoperability is **modular design** which is also a basic tenet of good development practice. Modularity will support the dynamic redeployment of a **Component** into different Nodes that requires the capabilities of the Component thus promoting broader interoperability between different Nodes and Components. Where possible, Nodes should adopt standards based, platform independent frameworks that facilitate ***pluggable*** deployment capabilities for Components so it can leverage the capabilities developed elsewhere.

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Client Platform
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange

Evaluation Criteria:

1) Test:

Does the Node support the elements of a modern component based framework such as **Java Platform, Enterprise Edition** (**Java EE**), **.NET** or **CORBA**?

Procedure:

Look for the existence of Java Platform, Enterprise Edition (Java EE), .NET or CORBA frameworks with in the Node's Component list or in its delivered software.

Example:

None.

G1619

Configure clients with a Common Access Card (CAC) reader.

Rationale:

DoD Instruction 8520.2, *Public Key Infrastructure (PKI) and Public Key (PK) Enabling* [R1206], defines **Common Access Card** (CAC) applicability and scope, in part, as follows:

This Instruction applies to:... 2.4. All DoD unclassified and classified information systems including networks (e.g., Non-secure Internet Protocol (IP) Router Network , Secret Internet Protocol Router Network, Web servers, and e-mail systems. Excluded are Sensitive Compartmented Information, and information systems operated within the Department of Defense that fall under the authority of the Director of Central Intelligence Directive (DCID) 6/3 (reference (h)).

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Client Platform / Common Access Card (CAC) Reader

Evaluation Criteria:

1) Test:

Do all the **client** and **server** hardware come equipped with **Common Access Card** (CAC) Readers?

Procedure:

Review the hardware list and verify that all hardware comes with or has external CAC readers.

Example:

None.

G1622

Implement commercial off-the-shelf (COTS) software that protects against malicious code on each operating system in the Node in accordance with the Desktop Application Security Technical Implementation Guide (STIG).

Rationale:

The viral and worm assault on computing resources is major concern but is not strictly limited to DoD hardware and operating systems. It has become a ubiquitous, wide spread problem that spreads destruction indiscriminately. Since the problem is not strictly a DoD problem, **commercial off-the-shelf** (**COTS**) solutions are always being updated to meet the current threats and are essential in protecting the assets. All hardware platforms should employ virus and worm detection and removal software that is routinely run (especially on hardware the runs Microsoft products).

> ***Note:*** *For purposes of this guidance, anti virus software includes related update and maintenance capabilities typically available with such packages.*

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Host Information Assurance
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Computing Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Computing Infrastructure Integrity

Evaluation Criteria:

1) Test:

Do all hardware devices listed in the Node acquisition list have COTS licensed virus and worm detection software?

Procedure:

Review the Node acquisition list and make sure there is one license for each piece of computer hardware.

Example:

None.

2) Test:

Do all hardware devices listed in the Node acquisition list have COTS virus and worm detection software installed?

Procedure:

Review the prerequisites in the installation manual for virus and worm software.

Example:

None.

G1623

Implement personal firewall software on computers used for remote connectivity in accordance with the Desktop Applications, Network, and Enclave Security Technical Implementation Guides (STIGs).

Rationale:

All hardware that is plugged into a network is subject to attack by hackers. In addition to hardware **firewalls** that may be in place, every piece of hardware should be protected by a software firewall. This is especially important for forward deployed computers that may not have an external firewalls on the local network. Personal firewalls continuously monitor the activity on the local computer network interface and detect possible hostile attacks. The user has the discretion to block hostile attacks permanently or for a particular occasion. Since this problem is not restricted to DoD assets, **commercial off-the-shelf** (**COTS**) products are continuously updated to meet the latest threats and are essential in meeting these threats.

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Host Information Assurance
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Decentralized Operations and Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Inter-Network Connectivity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Computing Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Computing Infrastructure Integrity

Evaluation Criteria:

1) Test:

Do all the hardware devices listed in the Node acquisition list have COTS software firewall licensed software?

Procedure:

Review the Node acquisition list and make sure there is one license for each piece of computer hardware.

Example:

None.

2) Test:

Do all hardware devises listed in the Node acquisition list have COTS **firewall** software installed and is it enabled?

Procedure:

Review the prerequisites in the installation manual for firewall software.

Example:

None.

G1624

Install anti-spyware software on all Windows Desktop computers.

Rationale:

Spyware is a category of malicious software that can impact system operation in ways similar to virus and other intrusions. Extending the principles of protection against viruses and other intrusions to spyware is an essential activity to ensure stable system operation and security. Anti-spyware software is required on all Windows computers per the Windows Desktop Application Security Technical Implementation Guide (STIG).

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Host Information Assurance
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Do all the Windows Desktop computers listed in the Node acquisition list have COTS software anti-spyware licensed software?

Procedure:

Review the Node acquisition list and make sure there is one license for each piece of computer hardware.

Example:

None.

2) Test:

Do all Windows Desktops listed in the Node acquisition list have COTS anti-spyware software installed and is it enabled?

Procedure:

Review the prerequisites in the installation manual for anti-spyware software.

Example:

None.

G1625

Provide a commercial off-the-shelf Directory Service that all of the components of a Node can use.

Rationale:

A Directory Service is a service that stores information about objects on a computer network. Common objects stored by a Directory Service include network users, common resources (such as shares and printers), authentication and authorization information.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Directory Services

Evaluation Criteria:

1) Test:

Is an Open Source directory service going to be used?

Procedure:

Review the prerequisites in the installation manual for open source directory service software.

Example:

None.

2) Test:

Is there a COTS directory service listed in the Node acquisition list?

Procedure:

Review the Node acquisition list and make sure there is one license for a directory service.

Example:

None.

G1626

Identify which Core Enterprise Services (CES) capabilities the Node Components require.

Rationale:

A Node needs to determine the set of **Core Enterprise Services** (**CES**) its **components** will require in order to ensure efficient prioritization of activities and resources to provide those services. **NCES** has defined a set of common capabilities that help categorize types of services that may be required by a Node's components. Identification of the capabilities the components require will help the Node determine which services to implement.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the list of components that comprise the Node indicate which CES capabilities are required to deploy each Component?

Procedure:

Review the list of components and verify that they have indicated which CES capabilities are required to support the component.

Example:

None.

G1627

Identify the priority of each Core Enterprise Services (CES) capability the Node components require.

Rationale:

Identifying the priority of capabilities required by the Node's **Components** will assist the Node in allocation of scarce resources towards the delivery of **CES** in the Node and minimize risks during deployment of Components within the Node. Some capabilities are ***essential*** at getting a component Deployed at a Node. Some are essential for a particular component increment. With this information the Node can construct a schedule that supports the transition and evolution of the current federation of systems to the **Global Information Grid** (**GIG**) vision.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the list of components that comprise the Node indicate the priority of the CES capabilities either relative to each other or as of a date?

Procedure:

Review the list of components and verify that they have indicated what the priority of the CES capabilities either relative to each other or as of a date.

Example:

None.

G1629

Identify which Net-Centric Enterprise Services (NCES) capabilities the Node requires during deployment.

Rationale:

Relying on a high-bandwidth **Transmission Control Protocol/Internet Protocol** (**TCP/IP**) network connection is not a reality for many deployed Nodes. These Nodes will have to develop many of their own **CES** capabilities for use by their member **components** while deployed. When the Node is not deployed, it may rely on proxies to the **Net-Centric Enterprise Services** (**NCES**) services.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

Evaluation Criteria:

1) Test:

Does the Node have a list of **Net-Centric Enterprise Services** (**NCES**) capabilities that it depends on while deployed?

Procedure:

Review the Node's documents for a list of Net-Centric Enterprise Services (NCES)capabilities required by the Node while deployed.

Example:

None.

G1630

Comply with the applicable Global Information Grid (GIG) Key Interface Profiles (KIPs) for implemented Core Enterprise Services (CES) in the Node.

Rationale:

When a **CES** is implemented locally, use the **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) developed by **DISA** as the authoritative definition of the interfaces. This allows a **component** that is hosted by one Node to be hosted on another Node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES and Intermittent Availability
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES and Intermittent Availability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Key Interface Profile (KIP)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Key Interface Profile (KIP)

Evaluation Criteria:

1) Test:

Do all **CES** used locally within the Node implement the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for Core Enterprise Services (CES) implement Global Information Grid (GIG) Key Interface Profiles (KIPs) for that CES.

Example:

None.

G1631

Expose Core Enterprise Services (CES) that comply with the applicable Global Information Grid (GIG) Key Interface Profiles (KIPs) in all Node services proxies.

Rationale:

A Node may expose or control access to **Global Information Grid** (**GIG**) **CES** by using **proxies**. This allows a **Component** that is hosted by one Node to be hosted on another Node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES and Intermittent Availability
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES and Intermittent Availability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Key Interface Profile (KIP)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Key Interface Profile (KIP)

Evaluation Criteria:

1) Test:

Do all **CES proxies** locally defined within the Node expose CES using the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for CES proxies follow Key Interface Profiles (KIPs) for that Global Information Grid (GIG) KIP.

Example:

None.

G1632

Certify and accredit Nodes with all applicable DoD Information Assurance (IA) processes.

Rationale:

Nodes are part of the DoD **Global Information Grid** (GIG) and are consequently required to have DoD **Information Assurance** (IA) certification and accreditation. Details for certification and accreditation are specified in DoD Directive 8500.1, DoD Instruction 8500.2, DoD Directive 8580.1, and DoD Instruction 5200.40. Satisfaction of these requirements results in IA compliance verification of the Node.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Information Assurance (IA)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Information Assurance (IA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Net-Centric IA Posture and Continuity of Operations
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the Node have DoD **Information Assurance** (IA) certification and accreditation?

Procedure:

Ask to examine the certification and accreditation reports.

Example:

None.

G1633

Host only DoD Information Assurance (IA) certified and accredited Components.

Rationale:

Nodes that expose the external Node users to non-certified or non-accredited **Components** represent a risk to the stability of the entire Node network and can introduce interoperability issues between Nodes (and related Components).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Information Assurance (IA)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Information Assurance (IA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Net-Centric IA Posture and Continuity of Operations
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the Node have a plan to scan all Components on a routine basis?

Procedure:

Look for a plan and examine the results of the scan.

Example:

None.

G1634

Certify and accredit Components with all applicable DoD Information Assurance (IA) processes.

Rationale:

Each **Component** could theoretically be deployed on any Node. Therefore, it is the responsibility of the Component to be DoD **Information Assurance** (IA) certified and accredited.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Information Assurance (IA)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Information Assurance (IA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Net-Centric IA Posture and Continuity of Operations
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Are all the **Components** DoD **Information Assurance** (IA) certified and accredited?

Procedure:

Examine the certification and accreditation reports.

Example:

None.

G1635

Make Nodes that will be part of the Global Information Grid (GIG) consistent with the *GIG Integrated Architecture*.

Rationale:

The **Global Information Grid** (**GIG**) architecture describes the basic, high level architecture in which Nodes reside. It is an integrated architecture consisting of the various **DoDAF** views. It provides a common lexicon and defines a basic infrastructure for the performance of information exchanges with other GIG Nodes using the **GIG Enterprise Services** (**GES**) and the **Net-Centric Enterprise Services** (**NCES**). The GIG Integrated Architecture is available via the DoD Architecture Repository System (DARS), https://dars1.army.mil/ [user account and PKI certificate required for access].

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Integrated Architectures
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Integrated Architectures
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Are there **DoDAF** integrated architecture products defined for the Node that are consistent with the **GIG** Integrated Architecture?

Procedure:

Look for the occurrence of **Operational View** (**OV**), **Systems and Services View** (**SV**), **Technical Standards View** (**TV**) and **All Views** (**AV**).

Example:

None.

G1636

Comply with the Net-Centric Operations and Warfare Reference Model (NCOW RM).

Rationale:

> ***Note:*** *CJCSI 6212.01E removed the NCOW RM element of the Net-Ready Key Performance Parameter (NR-KPP), integrating the components of the former NCOW RM into other elements of the NR-KPP.*

The **Net-Centric Operations and Warfare Reference Model** (**NCOW RM**) focused on achieving net-centricity. Compliance with the NCOW RM translated to articulating how each Node approached and implemented net-centric features. Compliance did not require separate documentation; rather, it required that a Node address, within existing architecture, analysis, and program architecture documentation, the issues identified by using the model, and further, make explicit the path to net-centricity the program is taking.

Node compliance with the NCOW RM is demonstrated through inspection and analysis:

- Use of NCOW RM definitions and vocabulary;
- Incorporation of NCOW RM **Operational View** (**OV**) capabilities and services in the materiel solution;
- Incorporation of NCOW RM **Technical Standards View Information Technology** (**IT**) and **National Security Systems** (**NSS**) standards in the **TV** products developed for the materiel solution.

Compliance with the NCOW RM initially was a critical component of compliance with the **Net-Ready Key Performance Parameter** (**NR-KPP**).

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Net-Centric Operations and Warfare Reference Model (NCOW RM)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Net-Centric Operations and Warfare Reference Model (NCOW RM)

Evaluation Criteria:

1) Test:

Have the instructions in Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 3170.01 been used to check the Node for Net-Centric Operations and Warfare Reference Model (NCOW RM) compliance?

Procedure:

Check Node documentation.

Example:

2) Test:

Have the instructions in Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01 been used to check the Node for Net-Centric Operations and Warfare Reference Model (NCOW RM) compliance?

Procedure:

Check Node documentation.

Example:

3) Test:

Have the instructions in the Defense Acquisition University (DAU) Guidebook section 7.2.6 been used to check the Node for NCOW RM compliance?

Procedure:

Check Node documentation.

Example:

G1637

Make Node-implemented directory services comply with the directory services Global Information Grid (GIG) Key Interface Profiles (KIPs).

Rationale:

When directory services are implemented locally, use the **Global Information Grid** (**GIG**) **KIPs** developed by DISA as the authoritative definition of the interfaces. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do all directory services used locally within the Node implement the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for directory services implement Global Information Grid (GIG) Key Interface Profiles (KIPs) for that directory services.

Example:

None.

G1638

Comply with the directory services Global Information Grid (GIG) Key Interface Profiles (KIPs) in Node directory services proxies.

Rationale:

A Node may expose or control access to **Global Information Grid** (**GIG**) directory services by using **proxies**. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do all directory services **proxies** locally defined within the Node expose directory services using the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for directory services proxies follow Key Interface Profiles (KIPs) for that Global Information Grid (GIG) KIPs.

Example:

None.

G1639

Describe Components exposed by the Node as specified by the Service Definition Framework

Rationale:

The construction of registry entries is specified by the **Service Definition Framework** (SDF) documented in Net-Centric Implementation Directives (NCIDs) S300. The common Service Definition Framework that serves as the basis for adequately describing the offered **Component** service from both a provider's and consumer's perspective. It describes the contract between the Component service provider and the Component service consumer, and serves as the basis for a **Service Level Agreement** (SLA). The common service definition framework consists of elements that include interface, service level, security and implementation information.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Service Enablers / Service Discovery
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 4: Node Guidance / Services / Service Enablers / Service Discovery

Evaluation Criteria:

1) Test:

Is there a **Service Definition Framework** (SDF) available for each of the Components' Services exposed through the Node?

Procedure:

Look for a Service Definition Framework (SDF) for each Component service exposed through the Node.

Example:

None

G1640

Register Components exposed by the Node with the DISA-hosted registries.

Rationale:

The best way to for an exposed Node's **Component** service to be discovered is by being registered in the DISA registry. The DISA registry implementation uses **Universal Description, Discovery, Integration** (UDDI).

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Service Enablers / Service Discovery
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 4: Node Guidance / Services / Service Enablers / Service Discovery

Evaluation Criteria:

1) Test:

Is the exposed Node's Component's service registered in the DISA **Universal Description, Discovery, Integration** (UDDI) Registry?

Procedure:

Examine the DISA Universal Description, Discovery, Integration (UDDI) Registry and look for the exposed Node's Component's service.

Example:

None.

G1641

Comply with the Service Discovery Global Information Grid (GIG) Key Interface Profiles (KIPs) in Node-implemented Service Discovery (SD).

Rationale:

When a **Service Discovery** (**SD**) is implemented locally, use the **Global Information Grid** (**GIG**) KIPs developed by DISA as the authoritative definition of the interfaces. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Service Enablers / Service Discovery
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 4: Node Guidance / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Does the **Service Discovery** (**SD**) used locally within the Node implement the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for Service Discovery (SD) implement Global Information Grid (GIG) Key Interface Profiles (KIPs) for that Service Discovery.

Example:

None.

G1642

Comply with the Service Discovery Global Information Grid (GIG) Key Interface Profiles (KIPs) in Node Service Discovery (SD) proxies.

Rationale:

A Node may expose or control access to **Global Information Grid** (**GIG**) **Service Discovery** (**SD**) by using **proxies**. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Service Enablers / Service Discovery
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 4: Node Guidance / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)

Evaluation Criteria:

1) Test:

Do the **Service Discovery** (**SD**) **proxies** locally defined within the Node expose Service Discovery using the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for Service Discovery (SD) proxies follow KIPs for that Global Information Grid (GIG) Key Interface Profiles (KIPs).

Example:

None.

G1643

Comply with the Federated Search - Registration Web Service (RWS) Global Information Grid (GIG) Key Interface Profiles (KIPs) in Node implemented Federated Search - Registration Web Service (RWS).

Rationale:

When a **Federated Search** - **Registration Web Service** (**RWS**) is implemented locally, use the **Global Information Grid** (**GIG**) KIPs developed by **DISA** as the authoritative definition of the interfaces. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search

Evaluation Criteria:

1) Test:

Does a **Federated Search** - **Registration Web Service** (**RWS**) used locally within the Node implement the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for Federated Search - Registration Web Service (RWS) implement Global Information Grid (GIG) Key Interface Profiles (KIPs) for that Federated Search - Registration Web Service (RWS).

Example:

None.

G1644

Comply with the Federated Search - Search Web Service (SWS) Global Information Grid (GIG) Key Interface Profiles (KIPs) in Node implemented Federated Search - Search Web Service (SWS).

Rationale:

When a **Federated Search** - **Search Web Service** (**SWS**) is implemented locally, use the **Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**) developed by **DISA** as the authoritative definition of the interfaces. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does **Federated Search** - **Search Web Service** (**SWS**) used locally within the Node implement the applicable **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Verify that the interfaces for Federated Search - Search Web Service (SWS) implement Global Information Grid (GIG) Key Interface Profiles (KIPs) for that Federated Search - Search Web Service (SWS).

Example:

None.

G1645

Implement a local Content Discovery Service (CDS).

Rationale:

The node should implement the **Content Discovery Service** (**CDS**) as part of the node infrastructure to be shared among the **Components** hosted at the Node. A CDS will allow other Nodes and Components to find content within the node. The systems within the Node normally provide the content.

> ***Note:*** *If a Node is frequently disconnected, has intermittent connectivity, or is otherwise isolated, then hosting a local CDS might not be a practical solution for external content discovery and more effective means for internal discovery may be applicable.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the Node implement the **Content Discovery Service** (**CDS**) **Global Information Grid** (**GIG**) **Key Interface Profile** (**KIP**)?

Procedure:

Look for an implementation at the Node of the Content Discovery Service (CDS) Global Information Grid (GIG) Key Interface Profiles (KIPs).

Example:

None.

G1646

Comply with the directory services Global Information Grid (GIG) Key Interface Profiles (KIPs) in Node Federated Search Services proxies.

Rationale:

A Node may expose or control access to **Global Information Grid** (**GIG**) **Federated Search** Services by using **proxies**. This allows a **Component** that is hosted by one Node to be hosted on another node with a minimal impact.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Do all **Federated Search** Services **proxies** locally defined within the Node expose Federated Search Services using the applicable **Global Information Grid KIP**?

Procedure:

Verify that the interfaces for Federated Search Services proxies follow KIPs for that Global Information Grid (GIG) Key Interface Profiles (KIPs).

Example:

None.

G1647

Provide access to the Federated Search Services.

Rationale:

Content Discovery Service can search across a set of Content Discovery Services and yield an integrated result. The current approach to providing this service is to harness an existing capability termed ***Federated Search*** developed under the **Horizontal Fusion** (**HF**) program. The capability utilizes the **DoD Discovery Metadata Specification** (**DDMS**).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management

Evaluation Criteria:

1) Test:

Does the Node provide access to the **Federated Search** Service **Global Information Grid** (GIG) **Key Interface Profile** (KIP)?

Procedure:

Look for a proxy or an implementation that provides access to the ***Federated Search***

Example:

None.

G1652

Use DoD PKI X.509 certificates for servers.

Rationale:

Using a DoD PKI X.509 **server certificate** identifies the server as being trusted by the DoD and guarantees that the server's identity is legitimate.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Identity Management
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Identity Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges

Evaluation Criteria:

1) Test:

Is the server certificate a valid DoD PKI X.509 certificate that is non-expired?

Procedure:

Open the server certificate and check that it is trusted by a trusted DoD root certificate.

Example:

G1662

Follow the guidance provided in the Security Technical Implementation Guide (STIG) for Domain Name System (DNS) implementations.

Rationale:

As a fundamental common service on **IP**-based networks, **DNS** is often a focal point for network attackers. Following the **STIG** ensures alignment with DoD identified security practices and configurations. The STIG addresses implementation options such as the choice of basic DNS server types (primary, secondary, caching-only), use of a split-DNS design, location of servers in the network and relationship to other network components, secure administration, security of zone transfers, and initial configuration.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Do the Node's **DNS** services follow the **STIG** for DNS implementations?

Procedure:

Compare Node DNS services configuration with those recommended by the STIG.

Example:

None.

G1667

Implement Virtual Private Networks (VPNs) in accordance with the guidance provided in the Network Security Technical Implementation Guide (STIG).

Rationale:

Virtual Private Networks provide a means for Node access to users outside the security enclave. To Network **STIG** provides recommendations on how to configure VPNs for secure access.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Network Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Network Infrastructure Integrity
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Subnets and Overlay Networks / Virtual Private Networks (VPN)
NESI / Part 4: Node Guidance / Node Transport / Subnets and Overlay Networks / Virtual Private Networks (VPN)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Other Design Tenets

Evaluation Criteria:

1) Test:

Does the configuration of the Node's **VPN** servers follow the recommendations of the Network **STIG**?

Procedure:

Check VPN server configuration against recommended configurations in the Network STIG.

Example:

None.

G1713

Use an Operating Environment (OE) for all Software Communications Architecture (SCA) applications that includes middleware which adheres to the Minimum CORBA Specification version 1.0.

Rationale:

Using a CORBA provider that adheres to the minimum CORBA v1.0, specification improves the interoperability between SCA Operating Environments.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Communication Architecture
NESI / Part 5: Developer Guidance / Middleware / Software Communication Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: RF Acquisition
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the OE contain middleware that provides the services and capabilities of minimum CORBA?

Procedure:

Check for minimum CORBA compliance in the CORBA provider's documentation.

Example:

G1714

Develop Software Communications Architecture (SCA) applications to use only Operating Environment functionality defined by the SCA Application Environment Profile.

Rationale:

The SCA Application Environment Profile (AEP) is a subset of the Portable Operating System Interface (POSIX) specification. Functionality that is not part of the AEP is not guaranteed to be part of the operating environment. Applications that rely on functionality that is not part of the AEP will require changes to deploy or port to other SCA platforms.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Communication Architecture
NESI / Part 5: Developer Guidance / Middleware / Software Communication Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: RF Acquisition
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the SCA application use Operating Environment functions not defined by a Application Environment Profile?

Procedure:

Check to see that all Operating Environment calls in the SCA application are listed in an Application Environment Profile.

Example:

G1717

Use constants instead of hard-coded numbers for characteristics that may change throughout the lifetime of the model.

Rationale:

Constants increase the usefulness and lifetime of a design because the model can adapt to a variety of environments by postponing or modifying those parameters late in the design cycle. This makes the code more readable, maintainable and reusable.

> ***Note:*** *This practice has been adapted from Cohen, section 1.6.1.1.3.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / VHDL / VHDL Coding and Design
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Are there any characteristics that are susceptible to modification that are directly given a value?

Procedure:

Parse the code and look for hard-coded characteristics that are susceptible to change and consider replacing them with a constant.

Example:

None

G1718

Design circuits to be synchronous.

Rationale:

The preferred method of engineering today's digital ICs is based on a synchronous design. The main advantages of this are simplicity and reliability. Creating synchronous pieces of code increases interoperability and reusability when they are used with other synchronous modules.

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / VHDL / VHDL Synchronous Design
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Are all flip-flops clocked by the same, common clock signal?

Procedure:

Check to make sure a single external clock signal triggers the design to go from a well defined and stable state to the next one. On the active edge of the clock, all input and output signals and all internal nodes are stable in either the high or low state. Between two consecutive edges of the clock, the signals and nodes are allowed to change and may take any intermediate state.

Example:

None

G1719

Automate testbench error checking in VHDL development.

Rationale:

Manual verification is subject to human error and is time consuming. In addition, automation promotes increased maintainability, because it enables fast and reliable verification of a model when modifications are made.

Note: *This practice has been adapted from Cohen, section 11.1.1.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / VHDL / VHDL Testbench
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Composeability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Does the testbench automatically report success or failure for each sub-test that it runs through?

Procedure:

Run the testbench to see if it automatically reports successes or failures for each sub-test.

Example:

None

G1724

Develop XML documents to be well formed.

Rationale:

By W3C definition, XML documents must be well formed. However, documents that contain XML tags that are not well formed has no name and is often still referred to as an XML Document in common vernacular. Therefore, this guidance statements helps to clarify the need for well-formed documents. Well formed XML documents are those documents which have a proper XML syntax. This is essential if the XML is to be parsed using common, readily available open source and commercial XML parsers.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Syntax
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Syntax
NESI / Part 5: Developer Guidance / Data / XML / XML Syntax
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Can the XML Document be parsed using a common, readily available XML Parser?

Procedure:

Open the XML document in a browser such as Mozilla Firefox or Microsoft Internet Explorer or use the XML Validator available from the W3 Schools at: http://www.w3schools.com/xml/xml_validator.asp

Example:

None

G1725

Develop XML documents to be valid XML.

Rationale:

The content of a **valid** XML document conforms to a specific set of user-defined content rules contained in XML schemas. XML schemas describe data values correctness using predefined data types as base types and assigning values to the data type specific attributes of those data types. For example, if an element in a document is required to contain text that can be interpreted as being an integer numeric value, and instead contains: alphanumeric text such as "hello"; is empty; or has other elements in its content, then the document is considered not valid.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Instance Documents
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XML Validation
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XML Validation
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XML Validation
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the document validation tool indicate that the XML document is valid?

Procedure:

Use a validating parser and verify that the document is valid.

Example:

None.

G1726

Define XML Schemas using XML Schema Definition (XSD).

Rationale:

While it is possible to use **Document Type Definitions** (DTD) to convey much of the same information as the **XML Schema Definition** (XSD), XSDs have a several distinct advantages which are very useful in terms of interoperability. For example, DTDs do not capture domain or type range information very well (i.e. elevation in meters is from 0 to 12,000).

XML Schemas are a tremendous advancement over DTDs. Here are some of the reasons to use XSDs versus DTDs as delineated by Roger Costello in an XML tutorial (see the ***XML Schema Tutorial*** available at http://www.xfront.com):

- Enhanced datatypes support:
  - 44+ in XSDs versus 10 in DTDs
  - Support for user defined datatypes. For example, a user can define a new type based on the string type. Elements declared of this type must follow this specific pattern ddd-dddd, where d represents a numeric digit.
- Written using the same syntax as other XML instance documents. This means there is less to remember and more consistency with the same rules applying to all XML instance documents.
  XSDs support a limited Object-oriented (OO) paradigm. For example, new types can be derived from previously defined types with more or more stringent restrictions.
- Supports a kind of polymorphism where elements can be interchanged with parent or child elements. For example, a "Book" element can be substituted for the "Publication" element.
- Supports the definition of elements that are unordered collections or sets of other elements.
- Support for the identification of elements as part of a unique key.
- Support for elements that have the same name but different content
- Support for elements that have a null (i.e., nil) value.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Are XML schemas defined using XML Schema Definitions?

Procedure:

Verify that XML schemas are defined using W3C XML Schema Definitions rather than Document Type Definitions.

Example:

None.

G1727

Provide names for XML type definitions.

Rationale:

By naming type definitions in a schema, the type definitions can be reused in any number of other definitions. For example:

```
<xsd:complexType name="PointOfContact">
 <xsd:sequence>
   <xsd:element name="LastName" type="xsd:string"/>
   <xsd:element name="FirstName" type="xsd:string"/>
   <xsd:element name="MiddleName" type="xsd:string"/>
   <xsd:element name="NickName" type="xsd:string"/>
   <xsd:element name="PhoneNumber" type="xsd:string"/>
 </xsd:sequence>
</xsd:complexType>
```

Can be reused anywhere a Point-Of-Contact needs to used. For Example:

```
<xsd:complexType name="Project">
 <xsd:sequence>
   <xsd:element name="ProjectName" type="xsd:string"/>
   <xsd:element name="ProgramManager" type="PointOfContact"/>
   <xsd:element name="HardwareManager" type="PointOfContact"/>
   <xsd:element name="SoftwareManager" type="PointOfContact"/>
   <xsd:element name="ConfigurationManager" type="PointOfContact"/>
 </xsd:sequence>
</xsd:complexType>
```

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Do all simpleTypes have names associated with them?

Procedure:

Examine all the simpleType elements in the schema and verify that they have a name associated with them.

Example:

```
<xsd:simpleType name="PointOfContact">
   ...
</xsd:simpleType>
```

2) Test:

Do all complexTypes have names associated with them?

Procedure:

Examine all the complexType elements in the schema and verify that they have a name associated with them.

Example:

```
<xsd:complexType name="PointOfContact">
  ...
</xsd:complexType>
```

G1728

Define types for all XML elements.

Rationale:

There are two ways to associate the type-like information within an XML Schema. The first way is define an **XML element** as a global element of the schema element and the second is to define a complex or simple type. The first method violates G1727 and it does not support the clean separation of the definition of types from the use of the types.

By separating the definition of the types from the definition of the elements within structures, the types can be reused and are loosely coupled from any particular instance of the domain. The definitions of the type information can be maintained by a community that wishes to share the definition rather than any particular implementation or instance.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the schema define any elements that are defined using references to other elements that are not part of a substitutionGroup rather than types?

Procedure:

Look for the use of an element's ref attribute.

Example:

None.

G1729

Annotate XML type definitions.

Rationale:

Types in a schema represent a particular concept or aspect within a particular subject domain. Providing documentation about the type within the schema itself helps prevent disconnects between the documentation and the implementation as captured by the type definition.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Do all the types defined within a schema have annotation that describes the nuances of type?

Procedure:

Look for an annotation for each simple type and complex type defined in the schema.

Example:

The complex type warranty includes an annotation that describes the purpose of the type and any caveats on when/how to use it.

G1730

Follow an XML coding standard for defining schemas.

Rationale:

There are any number of coding standards that are defined for coding XML Schemas. Here are some areas covered by the most popular:

- Elements and Types are Upper Camel Case (UCC) convention.
- Type names end with the word Type.
- Attributes start with a lowercase letter and then revert to Lower Camel Case (LCC) convention.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is there a consistent XML coding convention followed when schemas are defined?

Procedure:

Look for the occurrence of a XML coding standard and verify that the XML Schemas follow the standard.

Example:

None.

G1731

Only reference XML elements defined by a Type in substitution groups.

Rationale:

The 35mm, disk, and 3x5 components are simply declared as standalone **XML elements** which may be substituted for the abstract RecordingMedium element.

Note: *All of these RecordingMedium components have a type that is the same as, or derived from, the RecordingMediumType.*

Note: *The abstract RecordingMedium is associated with a type, RecordingMediumType, rather than defining the structure as part of the RecordingMedium element. This allows the definition of the RecordingMedium structure (i.e. type) to evolve independently.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Do substitutionGroup references point to an abstract element that has a structures defined by a type?

Procedure:

Ensure that all substitutionGroups point to an abstract element that has a structures defined by a type.

Example:

None.

G1735

Use the `.xsd` file extension for files that contain XML Schema definitions.

Rationale:

It is possible to use any name for a schema file extension. However, using any extension other than **`.xsd`** causes confusion for humans as well as tools and utilities which rely on MIMEs often mapped to file extensions.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Is the file extension that contains the schema definition .xsd?

Procedure:

Make sure that all XML documents that contain the xml **`schema`** tag have a file extension of .xsd.

Example:

None.

G1736

Separate document schema definition and document instance into separate documents.

Rationale:

Separating the definition of the schema from the document instance supports the modularity by separating the definition of structure from the actual data. Each is allowed to evolve and change independently. In most cases, the definition of the structure of the data should be relatively static compared with the number of documents that are shared using that schema.

Document name: Camera.xsd

```
<xsd:schema
    targetNamespace="http://www.camera.org"
    elementFormDefault="qualified">
 <xsd:include schemaLocation="Nikon.xsd"/>
 <xsd:include schemaLocation="Olympus.xsd"/>
 <xsd:include schemaLocation="Pentax.xsd"/>
 <xsd:element name="Camera">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element
          name="Body"
          type="BodyType"/>
       <xsd:element
          name="Lens"
          type="LensType"/>
       <xsd:element
          name="ManualAdapter"
          type="ManualAdapterType"/>
     </xsd:sequence>
   </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

Document name: Camera.xml

```
<?xml version="1.0"?>
<Camera xmlns ="http://www.camera.org"

        xsi:schemaLocation=
                   "http://www.camera.org
                    Camera.xsd">
 <Body>
   <Description>
      Ergonomically designed casing for easy handling
   </ Description>
 </Body>
 <Lens>
   <Zoom>300mm</Zoom>
   <F-Stop>1.2</F-Stop>
 </Lens>
 <ManualAdapter>
   <speed>1/10,000 sec to 100 sec</speed>
 </ManualAdapter>
</Camera>
```

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / XML Schema Files

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / XML Schema Files
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Instance Documents
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Does the instance document have a <schema> tag?

Procedure:

Check the instance document and look for the use of the schema tag or the use of the XMLSchema namespace.

Example:

None.

G1737

Define a target namespace in schemas.

Rationale:

A target namespace describes the namespace for all the schema components defined by the schema. Without a target namespace, all enclosed schema components are not associated with a namespace and if a namespace prefix is not associated with the target namespace then all references to these schema components must be unqualified. By not specifying a target namespace, ambiguity can arise when the schema is integrated with other schemas. This can cause unnecessary naming collisions.

Note: *http://www.library.org is the target namespace as well the lib namespace. See the third targetNamespace line of the following code sample.*

```
<?xml version="1.0"?>
<xsd:schema
      targetNamespace="http://www.library.org"

      elementFormDefault="qualified">
 <xsd:include schemaLocation="BookCatalogue.xsd"/>
 <xsd:element name="Library">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="BookCatalogue">
         <xsd:complexType>
           <xsd:sequence>
             <xsd:element ref="lib:Book"
                           maxOccurs="unbounded"/>
           </xsd:sequence>
         </xsd:complexType>
       </xsd:element>
     </xsd:sequence>
   </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the schema declare a target namespace?

Procedure:

Check the definition of all schemas and look for the assignment of the targetNamespace attribute.

Example:

```
<xsd:schema

  targetNamespace="http://www.library.org"
 >
  . . .
</xsd:schema>
```

G1738

Define a qualified namespace for the target namespace.

Rationale:

To force all schema components defined by the schema to be qualified and to belong to a namespace, associate a qualified namespace with the target namespace. This causes all components defined within the namespace to be explicitly associated with a namespace. In other words, all components are always qualified.

Note: *http://www.library.org is the target namespace as well the lib namespace. See the forth xmlns:lib line of the following code sample.*

```
<?xml version="1.0"?>
<xsd:schema
      targetNamespace="http://www.library.org"

      elementFormDefault="qualified">
 <xsd:include schemaLocation="BookCatalogue.xsd"/>
 <xsd:element name="Library">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="BookCatalogue">
         <xsd:complexType>
           <xsd:sequence>
             <xsd:element ref="lib:Book"
                           maxOccurs="unbounded"/>
           </xsd:sequence>
         </xsd:complexType>
       </xsd:element>
     </xsd:sequence>
   </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the schema declare a qualified namespace for the target namespace?

Procedure:

Check the definition of all schemas and look for the assignment of the targetNamespace attribute and make sure there is also a qualified namespace with the same name.

Example:

In this example, the targetNamespace and the qualified namespace lib both have the same URI associated with them.

```
<xsd:schema

  targetNamespace="http://www.library.org"
 >
  . . .
</xsd:schema>
```

G1740

Append the suffix Type to XML type names.

Rationale:

Syntactically, XML allows names within a namespace to be reused as long as they do not define the same XML Schema component. Therefore, a type and an element can both have the same name. A parser can easily differentiate the components, but a human can not. In order to maintain maintainable "user-friendly" code, differentiate types and elements by adding a type suffix for types.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Do all the complex type names end in the type suffix?

Procedure:

Examine all the complex and simple type schema component definitions and verify that they end in the suffix type.

Example:

None.

G1744

Only reference abstract XML elements in substitution groups.

Rationale:

An abstract **XML element** can not have its type instantiated in an instance document. This means that the element used as the basis for the substitution group and all the members of the substitution group must be derived from the same type.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Is the element used as the basis for the substitution group declared to be abstract and is it derived from a type?

Procedure:

Examine all the elements used as the basis for substitution groups and verify that they have been declared as abstract.

Example:

```
<xsd:element name="RecordingMedium"
      abstract="true"
      type="RecordingMediumType"/>
```

G1745

Append the suffix Group to substitution group XML element names.

Rationale:

Syntactically, XML allows names within a namespace to be reused as long as they do not define the same XML Schema component. Therefore, a type and an **XML element** can both have the same name. A parser can easily differentiate the components, but a human can not. In order to maintain maintainable "user-friendly" code, differentiate types and elements by adding a type suffix for types.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Substitution Groups
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Do all the complex type names end in the type suffix?

Procedure:

Examine all the complex and simple type schema component definitions and verify that they end in the suffix type.

Example:

None.

G1746

Develop XSLT style sheets that are XSLT version agnostic.

Rationale:

There are never any guarantees as to the XSLT environment that a stylesheet will be used in. There are ways of writing code as recommended by the W3C so that the stylesheets operate in XSL Version 1.0, 2.0 and future releases. See W3C Extensibility and Fallback for XSL Transformations (XSLT) 2.0 for details.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the style sheet support version 1.0 and 2.0 portability as defined by the W3C Extensibility and Fallback for XSL Transformations (XSLT) 2.0?

Procedure:

Look for the use of the xsl:when and xsl:otherwise construct where the 2.0 functions are tested for availability in the xsl:when branch and the 1.0 functionality is defined in the xsl:otherwise branch. For a comprehensive list of 2.0 functions see the W3Schools site on XPath, XQuery and XSLT Functions.

Example:

```
<out xsl:version="2.0">
 <xsl:choose>
   <xsl:when
      test="function-available('matches')">
     <xsl:value-of
        select="matches($input, '[a-z]*')"/>
   </xsl:when>
   <xsl:otherwise>
     <xsl:value-of
        select=
          = "string-length
              ( translate
               ( $in,
                 'abcdefghijklmnopqrstuvwxyz',
                   ''
               )
              )
             = 0"
      />
   </xsl:otherwise>
 </xsl:choose>
</out>
```

2) Test:

Does the style sheet support 2.0 and future version portability as defined by the W3C Extensibility and Fallback for XSL Transformations (XSLT) 2.0?

Procedure:

Look for the use of the use-when attribute in the xsl:value element.

Example:

```
<xsl:value-of
   select="pad($input, 10)"
   use-when="function-available('pad', 2)"
/>
<xsl:value-of
  select
    ="concat
       ( $input,
         string-join
          ( for $i in
              1 to
              10 - string-length($input)
              return ' ',
           ''
          )
       )"
  use-when="not(function-available('pad', 2)
"/>
```

G1751

Document all XSLT code.

Rationale:

XSLT is source code and should be internally documented including a file header that describes the purpose of the transform and any restrictions or caveats associated with the transform.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Doe the XSLT have internal comments that document the transform?

Procedure:

Look inside the XSLT code and look for internal comments.

Example:

```
<xsl:for-each
  select="/transactions/transaction">
 <!--
    NOTE: Since dates are currently in
    ISO format they are in a sorted format
    and need no multi-level sorting
  -->
 <xsl:sort
    order="ascending"
    select="@startdate"/>
 <tr>
   <td>
     <xsl:value-of
        select="@startdate"/>
   </td>
   <td>
     <xsl:value-of
        select="@description"/>
   </td>
   <td>
     <!#  Get year
           1234567890
           yyyy/mm/dd
      -->
     <xsl:value-of
        select="substring(@startdate, 1,4)"
      />
   </td>
   <td>
     <!#  Get month
           1234567890
           yyyy/mm/dd
      -->
     <xsl:value-of
        select="substring(@startdate, 6,2)"/>
   </td>
   <td>
     <!#  Get day
           1234567890
```

```
            yyyy/mm/dd
       -->
      <xsl:value-of
         select="substring(@startdate, 9,2)"/>
    </td>
  </tr>
</xsl:for-each>
```

G1753

Declare the XML schema version with an XML attribute in the root XML element of the schema definition.

Rationale:

Formalizing the schema version number through the use of a required **XML attribute** helps automate the process of validating the versions. This will reduce unexpected runtime errors that occur when assumptions are made about the schema that may change over time. (See http://www.xfront.com/SchemaVersioning.html)

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Does the schema definition define a required attribute that captures the version information?

Procedure:

Look at the schema definition file and look for the inclusion of a required attribute that captures the schema version number. In the following example, the schemaVersion attribute is defined.

Example:

```
<xs:schema

  targetNamespace="http://www.exampleSchema"
  xmlns: xs ="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  version="1.3"
>
 <xs:element name="Example">
```

```
  <xs:complexType>
     . . .
    <xs:attribute
       name="schemaVersion"
       type="xs:decimal"
       use="required"
     />
  </xs:complexType>
</xs:element>
```

G1754

Give each new XML schema version a unique URL.

Rationale:

This allows the previous versions of the schema to be made available to support uninterrupted processing and supports an orderly transition. It also allows the users of the schemas to compare and contrast the evolving schema. http://www.xfront.com/SchemaVersioning.html

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Versioning XML Schemas
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Look for the multiple schemas that represent different versions with different URLs.

Procedure:

Look for XSDs that all define a particular schema but can be found at different locations. This can be done by changing the path to the schema definition or that change the name of the file by adding the version number.

Example:

Changing the file path:

```
http://www.some.org/schema/1999/CoiSchema
http://www.some.org/schema/2003/CoiSchema
http://www.some.org/schema/2006/CoiSchema
```

Changing the file name:

```
http://www.some.org/schema/CoiSchema_1999
http://www.some.org/schema/CoiSchema_2003
http://www.some.org/schema/CoiSchema_2006
```

G1755

Use accepted file extensions for all files that contain XSL code.

Rationale:

It is possible to use any name for an XSL file extension. However, using any extension other than xsl or XSLT causes confusion for humans as well as tools and utilities which rely on MIMEs often mapped to file extensions.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Is the file extension that contains the XSL files .xsl or .xslt?

Procedure:

Make sure that all XSL files have a file extension of .xsl or xslt.

Example:

None.

G1756

Isolate XPath expression statements into the configuration data.

Rationale:

XPath expression statements are dependent on the XML Schemas that are associated with the documents. Consequently they need maintained independently from the applications that use them. Storing the XPath expression statements externally as part of the configuration data ensures a clean separation of the maintenance tasks and supports traceability using configuration management tools.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XPath
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XPath
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XPath
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability

Evaluation Criteria:

1) Test:

Are there XPath expression statements embedded as string literals in the application source code?

Procedure:

Look for the occurrence of XPath expression statements or XML Element names defined as strings within the source code.

Example:

```
void main ( String args)
{ . . .
  String titleSearchExpression
    = "/library/books/book/title";
  . . .
} // End main
```

G1759

Use a style guide when developing Web portlets.

Rationale:

Portals contain portlets from different sources, and it is important for usability for the portal to have a common look and feel across all portlets.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Do all portlets comply with a style guide.

Procedure:

Look at development documentation to determine if a style guide exist for Web portlets and look for code reviews that show it was used during development.

Example:

None.

G1760

Solicit feedback from users on user interface usability problems.

Rationale:

Active testing and solicitation of input from users helps identify usability problems with the user interface and helps to identify areas that may reduce performance or require excessive cognitive attention by the user.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs

Evaluation Criteria:

1) Test:

Does the program solicit user feedback for user interface usability problems?

Procedure:

Determine if user surveys are conducted on the usability of the system.

Example:

G1761

Provide units of measurements when displaying data.

Rationale:

Displayed units for measurable data provide for better understanding the data and enable reuse of the data. (This guidance is derived from MIL-STD 1472F)

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the system display units for all measurable data?

Procedure:

Inspect the user interfaces for system and check that units are shown for all measurable data.

Example:

Length displayed as meters
Distance displayed as miles.

G1762

Indicate all simulated data as simulated.

Rationale:

Simulated data that is not marked as simulated may be of misinterpreted and can decrease system, user, or system safety. (This guidance is derived from MIL-STD 1472F)

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable

Evaluation Criteria:

1) Test:

Is all simulated data clearly marked as simulated?

Procedure:

Check system inputs and outputs including user interfaces and check that the simulated data is properly labeled as simulated.

Example:

None.

G1763

Indicate the security classification for all classified data.

Rationale:

Displaying classified data without clearing marking the classification can lead to incorrect assumptions about the data. This can lead to improperly use of the data or prevent the data from being reused due to lack of clear understanding of the classification. (This guidance is derived from MIL-STD 1472F)

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Trustable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the system display classification markings for all classified data?

Procedure:

Check the system outputs and user interfaces for classification marking for all classified data.

Example:

Classification banners on monitors and printouts.

G1770

Explicitly define Data Distribution Service (DDS) Domains.

Rationale:

DDS uses Domains to separate the **Global Data Spaces** into independent areas. **Topics** written to one DDS Domain are completely hidden from the other DDS Domains. Use DDS Domains for isolation (hiding subsystem data from other parts of the system), modularity, and scalability. In order for systems to benefit from these advantages, they must explicitly define their own DDS Domains rather than use the default DDS Domain.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture

Evaluation Criteria:

1) Test:

Is the system using different **`DomainId`** values to isolate the subsystems?

Procedure:

Look for multiple calls to **`create_participant()`** operation on the **`DomainParticipantFactory`**.

Example:

```
participantFactory
  = TheParticipantFactory;
quickQuoterParticipant
  = participantFactory->create_participant
      ( QUICK_QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
realtimeQuoterParticipant
  = participantFactory->create_participant
      ( REALTIME_QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111

G1771

Explicitly define the Data Distribution Service (DDS) Quality of Service (QoS) Policies to describe the behavior of a publisher.

Rationale:

DDS relies on the use of QoS characteristics to match publishers with **subscribers**. If the publishers do not specify a QoS policy other than the default, much of the power of DDS publishing is lost and the capabilities of the publisher are not documented.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the **`get_default_publisher_qos`** operation used to create publisher?

Procedure:

Look for the use of the **`get_default_publisher_qos`** operation within the code.

Example:

```
participant
  = participantFactory->create_participant
      ( QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
DDS::PublisherQos publisherQos;
Participant->get_default_publisher_qos
  ( publisherQos );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111

2) Test:

Are values other than the **`PUBLISHER_QOS_DEFAULT`** value used to create publishers?

Procedure:

Verify that the **`PUBLISHER_QOS_DEFAULT`** constant is not used within the code.

Example:

```
DDS::Publisher publisher
```

```
  = participant->create_publisher
      ( PUBLISHER_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111

G1772

Assign a unique identifier for each Data-Distribution Service (DDS) Domain.

Rationale:

DDS uses Domains to separate the **Global Data Spaces** into independent areas. Within DDS, a unique identifier called the **`DomainId`** identifies each DDS Domain.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / DDS Domains - Global Data Spaces
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is there a single value for the **`DomainId`** used for each Domain when the **`create_participant`** operation is used?

Procedure:

Look for the use of the **`create_participant`** operation within the code.

Example:

```
participantFactory
  = TheParticipantFactory;
quickQuoterParticipant
  = participantFactory->create_participant
      ( QUICK_QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
realtimeQuoterParticipant
  = participantFactory->create_participant
      ( REALTIME_QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111

G1773

Use `#include` guards for all headers.

Rationale:

Including a guard prevents including a header file more than once. There are two basic kinds of guards: internal and external. Internal guards occur in each header file that is to be included. External guards occur in a file that includes a header file. In the past, there were compiling performance issues using internal guards because the file had to be scanned each time the file was included. This has been optimized away by most modern compilers. Furthermore, external guards are fragile and tightly coupled since the file including the header and header file must use the same guard name.

Note: *This practice has been adapted from Sutter and Alexandrescu, standard practice 24.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Header Files
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Do all header files contain include guards?

Procedure:

Check each file that is included using a **`#include`** statement to make sure it has an include guard.

Example:

An internal guard looks like this:

```
#ifndef MYHEADER_HPP

#define MYHEADER_HPP

... // Contents of include file go here

#endif
```

G1774

Make header files self-sufficient.

Rationale:

To enable code reuse, each unit of code should be able to be compiled independently without having to follow a predetermined build order or having to know the dependencies. Code is difficult to reuse when the dependencies are not clearly documented. Therefore, ensure each header is capable of being used by itself (i.e., it can be compiled standalone) by having it include all the headers upon which it depends.

Note: *This practice has been adapted from Sutter and Alexandrescu, standard practice 23.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Header Files
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Can each class be compiled by itself without having to compile other units?

Procedure:

Compile each class as a standalone file and check compile output for errors caused by missing definitions.

Example:

None

G1775

Do not overload the logical `AND` operator.

Rationale:

The logical **`AND`** operator has a special relationship with the compiler. When a logical **`AND`** operator is written to overload the inherent operators, the precedence of operation (i.e., left side of operator or right side of operator) is undefined. This can result in compiler dependency. In the following code, it is not clear whether the **`DisplayPrompt`** will execute first or the **`GetLine`** operation will executed first.

```
if ( DisplyPrompt() && GetLine() )
```

Note: *This practice has been adapted from Sutter and Alexandrescu, standard practice 30.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Operator Overloading
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Is the logical **`AND`** operator defined?

Procedure:

Look for the overloading of the logical **`AND`** operator.

Example:

None

G1776

Do not overload the logical `OR` operator.

Rationale:

The logical **`OR`** operator has a special relationship with the compiler. When a logical **`OR`** operator is written to overload the inherent operators, the precedence of operation (i.e., left side of operator or right side of operator) is undefined. This can result in compiler dependency.

> ***Note:*** *This practice has been adapted from Sutter and Alexandrescu, standard practice 30.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Operator Overloading
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Is the logical **`OR`** operator defined?

Procedure:

Look for the overloading of the logical **`OR`** operator.

Example:

None

G1777

Do not overload the `comma` operator.

Rationale:

The **`comma`** operator has a special relationship with the compiler. When a **`comma`** operator is written to overload the inherent operators, the precedence of operation (i.e., left side of operator or right side of operator) is undefined. This can result in compiler dependency.

> ***Note:*** *This practice has been adapted from Sutter and Alexandrescu, standard practice 30.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Operator Overloading
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Is the **`comma`** operator defined?

Procedure:

Look for the overloading of the **`comma`** operator.

Example:

None

G1778

Place all `#include` statements before all namespace `using` statements.

Rationale:

Files that are included can contain their own **`using`** clauses. In order to make sure that the **`using`** statements are not overridden by these subsequent using definitions, place all using statements after all include statements.

> ***Note:*** *This practice has been adapted from Sutter and Alexandrescu, standard practice 59.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Namespaces and Modules
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Are all the **`using`** statements defined after all the **`#include`** statements?

Procedure:

Scan all files and make sure that all the **`using`** statements occur after all **`using`** statements.

Example:

None

G1779

Explicitly namespace-qualify all names in header files.

Rationale:

Header files are meant to be included by other files. A header file inclusion should not alter the meaning of code that it is included in as this behavior is unexpected. Therefore, use fully-qualified names in header files and do not use using directives or declarations. This also promotes clarity in the header file whose main purpose is to communicate the interface to the implementation class.

> ***Note:*** *This practice has been adapted from Sutter and Alexandrescu, standard practice 59.*

Referenced By:

NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Namespaces and Modules
NESI / Part 5: Developer Guidance / Programming Languages / C++ / C++ Header Files
NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Are named fully namespace qualified throughout the header files?

Procedure:

Scan all header files and make sure that all namespaces are fully qualified.

Example:

None

2) Test:

Are all header files free from using directives or declarations?

Procedure:

Scan all header files to determine that they do not contain using directives or declarations.

Example:

None

G1784

Include a statement in the solicitation for Contractors to identify and list data rights for all proposed products.

Rationale:

Reusing GOTS requires understanding all the data rights associated with each artifact involved with the solution.

Referenced By:

NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Section K: Representations, Certifications, and Other Statements of Offerors (Data Rights)
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Does the solicitation include a statement for the offerer to identify data rights for all proposed products?

Procedure:

Review the solicitation and identify statements that require the offerer to identity data rights for all proposed products.

Example:

Example data rights markings include markings for Unlimited Rights and Government Purpose Rights.

G1785

Stipulate that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on reuse of common functionality.

Rationale:

The Government must stipulate what evaluation criteria will be used to evaluate proposed solutions. Having the Offeror specify the extent to which proposed solutions build on reuse of common functionality aids in the evaluation of proposals and aids in identification of common functionality.

Referenced By:

NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Section M: Evaluation Factors for Award
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Has the government stipulated that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on reuse of common functionality?

Procedure:

Check Section M for a statement that states reuse of common functionality will be used as an evaluation criterion for proposals.

Example:

None.

G1786

Stipulate that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on well defined services.

Rationale:

The Government must stipulate what evaluation criteria will be used to evaluate proposed solutions. Having the Offeror specify the extent to which proposed solutions build on reuse of well defined services aids in the evaluation of proposals and further improves service reuse.

Referenced By:

NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Section M: Evaluation Factors for Award
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Has the government stipulated that evaluation criteria will include the extent to which an Offeror's proposed technical solution builds on well defined services?

Procedure:

Check Section M for a statement that states the extent to which the proposed solution builds on well defined services will be used as an evaluation criterion for proposals.

Example:

None.

G1787

Stipulate that the Offeror is to use the NESI *Net-Centric Implementation* documentation set to assess net-centric interoperability.

Rationale:

NESI guidance and its associated checklists are useful tools (used by themselves or in conjunction with other tools) for assessing how a program is meeting its net-centric and interoperability objectives.

Referenced By:

NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Section J: List of Attachments
NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Post Award Contract Actions
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Has the Government stipulated that the Offeror is to use NESI to assess net-centricity and interoperability?

Procedure:

Identify statements in policy, RFPs, SOWs, or CDRLs that stipulate that the Offeror is to use NESI to assess net-centricity and interoperability?

Example:

PEO C4I uses the Technical Evaluation Checklist (http://nesipublic.spawar.navy.mil/checklist) as a means for Program Managers to assess how well their programs meet net-centric objectives.

G1788

Stipulate that the Offeror is to use Government approved data rights labels and markings for all deliverables that are identified as Unlimited or Government Purpose Rights.

Rationale:

Reusing deliverables or components of deliverables requires a full understanding of the data rights associated with each artifact in the deliverable. Identified data rights for each artifact through the use of data right labels are important in order to protect the legal rights of both the contractor and government during component reuse.

Referenced By:

NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Section J: List of Attachments
NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Post Award Contract Actions
NESI / Part 2: Traceability / Naval Open Architecture / Reusability

Evaluation Criteria:

1) Test:

Has the government stipulated that the Offeror is to use government approved data rights labels and markings for all deliverables that are identified as Unlimited or Government Purpose Rights.

Procedure:

Identify statements in the RFP, SOW, or CDRLs which mandate the use of government approved data rights labels for any deliverables that are identified as Unlimited or Government Purpose Rights.

Example:

None.

G1796

Explicitly define Data Distribution Service (DDS) Domain Topics.

Rationale:

DDS uses Topics to define the information model. Topics are identified by an application-defined string and an associated **data type**. Topics represent collections of object sin the **Global Data Space**; individual data-objects within a Topic are identified by the value of the key fields which are some special fields inside the data-type. Applications use Topics to publish the information and subscribe to the information they want.

In a DDS system information exchange happens as a result of **publishers** and **subscribers** agreeing to use the same Topics. Therefore the selection of the Topic names and their semantic meaning is an important part of system design.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Are all the Topics (and Topic names) explicitly defined and captured in a publicized data source (e.g., Excel table, XML file, dedicated tool)?

Procedure:

Look for documentation that contains listings for all Topics the system uses.

Example:

```
<topic>
 <name>Temperature</name>
 <type>TemperatureData</type>
 <description>
   This topic contains a reading of
   a temperature sensor
 </description>
</topic>
<topic>
  . . .
</topic>
```

G1797

Use a minimum of 1024 bits for asymmetric keys.

Rationale:

Strong encryption helps to prevent unauthorized data decryption using modern day resources.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Encryption Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE

Evaluation Criteria:

1) Test:

Are asymmetric key encryption levels at least 1024 bit?

Procedure:

Check the server configuration and verify that the asymmetric keys being used are at least 1024 bit.

Example:

Verified Web server ciphers under the SSL portion of the configuration pages of the administration server.
For Internet Explorer 5.0 and above, click the **`Help`** menu and then click the **`About Internet Explorer`** option. The **`About`** box will list the Cipher Strength.

2) Test:

Is the application using domestic (U.S.) grade ciphers?

Procedure:

Verify that the application supports domestic (U.S.) grade ciphers.

Example:

None.

G1798

Explicitly define all the Data Distribution Service (DDS) Domain data types.

Rationale:

DDS provides support for writing and reading typed data. For each application data type, DDS creates the necessary objects that allow manipulation of the data object. For example, for a given data type named **`MyDT`**, DDS creates a **`MyDTDataWriter`** and **`MyDTDataReader`**.

Knowing the data type of the object allows DDS to marshal the data properly. Consequently, any computer platform and/or language can process the data properly . For example, DDS performs the proper endianess transformations, alignment, and adjustment for 32 versus 64 bit platforms.
Knowing the data type is also required for the proper functioning of **`ContentFilteredTopics`**.

Moreover, explicit definition of the data types is required for the tools provided by DDS vendors to display and manipulate the data properly. Visualization tools, logging and replay, automatic bridging to other middleware, etc., all depend on data type transparency.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Are all the data types the system uses explicitly defined using IDL which is either manually written or generated from equivalent UML or XML representations?

Procedure:

Look for the IDL (or equivalent XML) files used to define the types used by the system.

Example:

```
// File MyTpes.idl
struct MyType
{
   long x;
   long y;
   string<10> units;
};
```

G1799

Explicitly associate data types to the Data Distribution Service (DDS) Topics within a DDS Domain

Rationale:

A DDS Topic represents a homogeneous collection of data-objects in the **Global Data Space**. All data-objects within a Topic share a common **data-type**. Knowledge of the type associated with the Topic is required for an application to be able to publish and subscribe data on the Topic.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Do all Topics have an explicit association to a data type.

Procedure:

Look for documentation that lists the Topics in use by the system and verify that each Topic has a data type associated with it

Example:

```
<topic>
 <name>Temperature</name>
 <type>TemperatureData</type>
 <description>
   This topic contains a reading of
   a temperature sensor
 </description>
</topic>
<topic>
  . . .
</topic>
```

G1800

Explicitly identify Keys within the Data Distribution Service (DDS) data type that uniquely identify an instance of a data object.

Rationale:

Within each DDS **Domain** (i.e., **Global Data Space**) a data-object is identified by the tuple (**Topic**, Key). The Key is a set of fields within the data type associated with the Topic that the application has tagged to indicate their role in uniquely identifying the data object. For example, if the Topic represents a person to the IRS, the Key may be simply the field containing the social security number.

The proper definition of the key is necessary to allow DDS to implement the **`KEEP_LAST HISTORY`** QoS properly as well as to enforce QoS policies such as **`DEADLINE`**, and **`OWNERSHIP`**. It is also necessary in order for DDS to supply the proper Sample information to the **DataReader**.

All data types require Keys except in the case where the Topic logically represents a single object, for example when the Topic represents a Message Queue.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Does the declaration of the data-type associated with the Topic explicitly designate using one or more of the fields as a Key?

Procedure:

Examine the IDL (or equivalent XML) files used to define the types used by the system to identify the declaration of the data-type associated with each Topic (i.e., see if there are any tags that designate which fields form the Key).

Example:

```
For data types defined using IDL:
struct SensorData
{
  long    sensor_id; //@key
  float   value;
  string<32> units;
  string<64> location;
};
struct DepartingFlightData
{
  string<8>    airline_code;  //@key
  long         flight_number; //@key
```

```
  string<8>    destination_airport_code;
  string<2>    departing_terminal;
  long         departing_gate;
  FlightTime   scheduled_departure_time;
  FlightTime   expected_departure_time;
  string<32>   status;
};
```

G1801

Explicitly define a Topic Quality of Service (QoS) for each Data Distribution Service (DDS) Topic within a DDS Domain.

Rationale:

DDS Topics define the information model of the system. The QoS Policies associated with the Topics define expectations and constraints that all users (**publishers** or **subscribers**) of the Topic should know. Consequently, definition of the Topic QoS is an important part of the system design.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Messaging within a DDS Domain
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is there a document that defines the QoS Policies that each Topic uses and does the document that describes the Topics and their associated data types also provide information on the Topic QoS?

Procedure:

Look at the documents that define the Topics in use and their associated data-types and see if they also define the Topic QoS.

Example:

```
Topic: DepartingAircraft
Type: DepartingAircraftStruct
QoS: HISTORY kind=KEEP_LAST
QoS: RELIABILITY kind=RELIABLE
QoS: DEADLINE duration=15minutes
QoS: LIFESPAN duration = 1 hour
Etc.
```

G1802

Catch Data Distribution Service (DDS) events.

Rationale:

DDS uses **listeners** to notify the application of relevant events such as mis-matched Topic definitions, **QoS** violations, lost samples, etc. Normally these events are dispatched to the most specific entity to which they apply (e.g., the affected **DataReader** in the case of the lost sample notification). However under application control the **`DataReader`** can "mask" certain events such that they are propagated to the enclosing container entity (e.g. the **Subscriber** to which the affected **`DataReader`** belongs). The **DomainParticipant** is the ultimate container of all DDS entities and it is therefore important that it handles (e.g., logs) any events that the contained entities have not handled.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

Evaluation Criteria:

1) Test:

Is a non-nil listener specified when the **`DomainParticipant`** is created?

Procedure:

Look at the arguments passed to the **`create_domain_participant`** operation on the **`DomainParticipantFactory`** and check the values of the listener and mask arguments.

Example:

```
participantFactory
  = TheParticipantFactory;
participant
  = participantFactory->create_participant
      ( QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

G1803

Explicitly define the Data Distribution Service (DDS) Quality of Service (QoS) Policies to describe real-time messaging criteria for Publishers.

Rationale:

DDS relies on the use of a QoS set of characteristics to match publishers with **subscribers**. If the publishers do not specify a QoS policy other than the default, much of the power of DDS publishing is lost and the capabilities of the publisher are not documented.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the **`get_default_publisher_qos`** operation used to create publisher?

Procedure:

Look for the use of the **`get_default_publisher_qos`** operation within the code.

Example:

```
participant
  = participantFactory->create_participant
      ( QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
DDS::PublisherQos publisherQos;
Participant->get_default_publisher_qos
  ( publisherQos );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

2) Test:

Is the **`PUBLISHER_QOS_DEFAULT`** value used to create publishers?

Procedure:

Look for the use of the **`PUBLISHER_QOS_DEFAULT`** constant within the code.

Example:

```
DDS::Publisher publisher
  = participant->create_publisher
```

```
      ( PUBLISHER_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

G1804

Explicitly define the Data Distribution Service (DDS) Quality of Service (QoS) Policies to describe DataWriter.

Rationale:

DDS relies on the use of QoS characteristics to match a **`DataWriter`** with each **DataReader** of the same **Topic**. If the **`DataWriter`** does not specify a QoS policy other than the default, much of the power of DDS publishing is lost and the capabilities of the **`DataWriter`** are not documented.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the **`get_default_datawriter_qos`** operation used to create participant?

Procedure:

Look for the use of the **`get_default_datawriter_qos`** operation within the code.

Example:

```
DDS::DataWriterQos dataWriterQos;
publisher->get_default_datawriter_qos
  ( dataWriterQos );
DDS::DataWriter dataWriter
  = publisher ->create_datawriter
      ( myTopic,
        dataWriterQos,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

2) Test:

Is the **`DATAWRITER_QOS_DEFAULT`** value used to create **`DataWriter?`**

Procedure:

Look for the use of the **`DATAWRITER_QOS_DEFAULT`** constant within the code.

Example:

```
DDS::DataWriter dataWriter
  = participant->create_datawriter
      ( myTopic,
```

```
        DATAWRITER_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

DDS::STATUS_MASK_ALL is part of DDS 1.3, prior releases require application to use 0x11111111.

G1805

Explicitly define the Data Distribution Service (DDS) Quality of Service (QoS) Policies to describe the behavior of the Subscriber.

Rationale:

DDS relies on the use of QoS set of characteristics to match subscribers with **publishers**. If the subscribers do not specify a QoS policy other than the default, much of the power of DDS subscription and publishing is lost and the requirements of the subscriber are not documented.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the **`SUBSCRIBER_QOS_DEFAULT`** value used to create subscribers?

Procedure:

Look for the use of the **`SUBSCRIBER_QOS_DEFAULT`** constant within the code.

Example:

```
DDS::Publisher publisher
  = participant->create_subscriber
      ( SUBSCRIBER_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

2) Test:

Is the **`get_default_subscriber_qos`** operation used to create subscribers?

Procedure:

Look for the use of the **`get_default_subscriber_qos`** operation within the code.

Example:

```
participant
  = participantFactory->create_participant
      ( QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
```

```
      );
DDS::SubscriberQos subscriberQos;
Participant->get_default_subscriber_qos
  ( subscriberQos );
```

DDS::STATUS_MASK_ALL is part of DDS 1.3, prior releases require application to use 0x11111111.

G1806

Explicitly define the Request-Offered Data Distribution Service (DDS) Quality of Service (QoS) Policies to describe the behavior of the DataReader.

Rationale:

DDS relies on the use of QoS characteristics to match a **DataWriter** with each **`DataReader`** of the same Topic. If the **`DataReader`** does not specify a QoS policy other than the default, much of the power of DDS subscription and publishing is lost and the requirements of the **`DataReader`** are not documented.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Is the **`DATAREADER_QOS_DEFAULT`** value used to create **`DataReader`**?

Procedure:

Look for the use of the **`DATAREADER_QOS_DEFAULT`** constant within the code.

Example:

```
DDS::DataResder dataReader
  = participant->create_datareader
      ( DATAREADER_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

2) Test:

Is the **`get_default_datareader_qos`** operation used to create participant?

Procedure:

Look for the use of the **`get_default_datareader_qos`** operation within the code.

Example:

```
DDS::DataReaderQos dataReaderQos;
publisher->get_default_datareader_qos
  ( dataReaderQos );
DDS::DataReader dataReader
  = publisher ->create_datareader
      ( myTopic,
```

```
        dataReaderQos,
        NULL,
        DDS::STATUS_MASK_ALL
      );
```

G1807

Check the return values of Data Distribution Service (DDS) functions.

Rationale:

Many of the DDS operations return a nil value when the operation does not work. Not checking for these **`nil`** values can cause unexpected and potentially non-deterministic behavior. Different implementations of the DDS may even behave differently when these values are used. The following is a list of operations that can return **`nil`**:

- create_publisher
- create_subscriber
- create_topic
- create_contentFilteredtopic
- create_multitoic
- find_topic
- lookup_topicdescription
- create_participant
- lookup_participant
- create_datawriter
- lookup_datawriter
- create_datareader
- lookup_datareader
- create_readcondition
- create_querycondition

One operation returns **`HANDLE_NIL`** when the operation fails.

- **`lookup_instance`**

The remaining operations return a **`DDS::ReturnCode_t`** enumerated value that indicates whether the operation succeeded (**`DDS::RETCODE_OK`**) of else the reason for failure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

Evaluation Criteria:

1) Test:

Do all invocations of the DDS operations lookup_instance check for a return value of **`HANDLE_NIL`**?

Procedure:

Examine the code for the use of the lookup_instance operations and make sure they check for the return of a **`DDS::HANDLE_NIL`** value immediately after the operation.

Example:

```
DDS::InstanceHandle_t instanceHandle
  = DDS::HANDLE_NIL;
instanceHandle
  = writer->lookup_instance( instance )
if ( instanceHandle == DDS::HANDLE_NIL )
  { cerr << "..."
        << endl;
    exit(1);
  } // End if
```

2) Test:

Are all of the DDS operations that can return **`nil`** values checked for the return of a **`nil`** values?

Procedure:

Examine the code for the use of the following operations and make sure they check for the return of a **`nil`** value immediately after the operation.

- create_publisher
- create_subscriber
- create_topic
- create_contentFilteredtopic
- create_multitoic
- find_topic
- lookup_topicdescription
- create_participant
- lookup_participant
- create_datawriter
- lookup_datawriter
- create_datareader
- lookup_datareader
- create_readcondition
- create_querycondition

Note: *Examine the return of any other operation and make sure they check for* ***`DDS:RETCODE_OK`*** *immediately after the operation.*

Example:

```
DDS::Publisher publisher
  = participant->create_publisher
      ( PUBLISHER_QOS_DEFAULT,
        NULL,
        DDS::STATUS_MASK_ALL
      );
if ( publisher == NULL ) )
  { cerr << "create_publisher failed."
        << endl;
    exit(1);
  } // End if
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

3) Test:

Are all invocations to DDS operations that return a **`DDS::ReturnCode_t`** checked for **`DDS::RETCODE_OK`**?

Procedure:

Examine the code for the use of the operations with prototype returning **`DDS::ReturnCode_t`** to make sure they check for the return of a **`DDS::RETCODE_OK`** immediately after the operation.

Example:

```
retcode
  = writer->write( # )
if ( retcode != DDS::RETCODE_OK )
  { cerr << "..."
        << endl;
    //  handle error
} // End if
```

G1808

Handle all Data Distribution Service (DDS) Quality of Service (QoS) contract violations using one of the Subscriber access APIs.

Rationale:

QoS contract violations typically indicate either a system mis-configuration, or else a transient failure (e.g., a network that has been temporarily disconnected). Either way the application must monitor these events to determine if they are relevant to their operation and consequently take proper corrective action.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability

Evaluation Criteria:

1) Test:

Are all the DDS QoS-related status change events are captured via a DDS **Listener** or a DDS **WaitSet**?

Procedure:

Specifically ensure that the following DDS events are handled. Look at the arguments passed to the **`create_domain_participant`**, **`create_datawriter`**, and **`create_datareader_operations`** and check that the listener and mask parameters to verify that the following events are being handled:

- OFFERED_DEADLINE_MISSED_STATUS
- REQUESTED_DEADLINE_MISSED_STATUS
- OFFERED_INCOMPATIBLE_QOS_STATUS
- REQUESTED_INCOMPATIBLE_QOS_STATUS
- LIVELINESS_LOST_STATUS
- LIVELINESS_CHANGED_STATUS

Example:

```
participantFactory
  = TheParticipantFactory;
quickQuoterParticipant
  = participantFactory->create_participant
      ( QUICK_QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        participantListener,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

G1809

Handle all Data Distribution Service (DDS) events using one of the subscriber access APIs.

Rationale:

Listeners and the dual **Condition**/**WaitSet** infrastructure allow applications to be notified when changes occur in a **DCPS** communication.

Listeners provide a generic mechanism for the middleware to notify the application of relevant asynchronous events, such as arrival of data corresponding to a **subscription**, violation of a **QoS** setting, etc. Each DCPS entity supports its own specialized kind of listener. Listeners are related to changes in status conditions. Listener operations are invoked using a middleware-provided thread.

Conditions and **`WaitSets`** provide the means for an application thread to block waiting for the same events that can be received via a Listener. Using a **`WaitSet`**, the application can handle the event in its own thread instead of the middleware provided thread used for Listeners.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

Evaluation Criteria:

1) Test:

Are all DDS status change events are captured via a DDS Listener or a DDS WaitSet?

Procedure:

Verify that the following DDS events are handled. Look at the arguments passed to the **`create_domain_participant`**, **`create_datawriter`**, and **`create_datareader_operations`** checking that the listener and mask parameters to verify that the following events are handled:

- **`INCONSISTENT_TOPIC_STATUS`**
- **`SAMPLE_LOST_STATUS`**
- **`SAMPLE_REJECTED_STATUS`**
- **`DATA_ON_READERS_STATUS`**
- **`DATA_AVAILABLE_STATUS`**
- **`OFFERED_DEADLINE_MISSED_STATUS`**
- **`REQUESTED_DEADLINE_MISSED_STATUS`**
- **`OFFERED_INCOMPATIBLE_QOS_STATUS`**
- **`REQUESTED_INCOMPATIBLE_QOS_STATUS`**
- **`LIVELINESS_LOST_STATUS`**
- **`LIVELINESS_CHANGED_STATUS`**

Example:

```
participantFactory
  = TheParticipantFactory;
quickQuoterParticipant
  = participantFactory->create_participant
      ( QUICK_QUOTER_DOMAIN_ID,
        PARTICIPANT_QOS_DEFAULT,
        participantListener,
        DDS::STATUS_MASK_ALL
      );
```

`DDS::STATUS_MASK_ALL` is part of DDS 1.3, prior releases require application to use 0x11111111.

G1810

Use data models to document the data contained within the Data Distribution Service (DDS) Data-Centric Publish Subscribe (DCPS).

Rationale:

DCPS contains static and raw data that can be used is any number of views or objects. As a consequence, changes in the definition of the data, its DDS **Domains** or its structure can have a huge cascading effect. To minimize the impact of these changes, data needs to be documented in a data model that is not subject to implementation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain

Evaluation Criteria:

1) Test:

Is there a conceptual data model that captures the data within the DCPS?

Procedure:

Look for a data model that captures the data within the Data-Centric Publish-Subscribe (DCPS). The following is a very short list of some of the files extensions that may contain data models.

CDM	Conceptual model file (PowerDesigner)
PDM	Physical model file (PowerDesigner)
ER1	ERWin file
ERX	ERWin file

ERM	Entity Relationship Diagram Model file (Prosa)

Example:

None.

G1862

Configure Active Directory for Smart Card Logon.

Rationale:

This is a DoD requirement; DoD Instruction 8520.2 [R1206] and DoD Directive 8190.3 [R1297] refer and Joint Task Force-Global Network Operations (JTF-GNO) Communications Tasking Order (CTO 06-02) specifically directs implementation of Smart Card Logon (SCL) on all **NIPRNet** networks.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Smart Card Logon
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Smart Card Logon
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Smart Card Logon

Evaluation Criteria:

1) Test:

Is Active Directory configured for SCL?

Procedure:

Verify that Active Directory is configured for SCL?

Example:

None.

G1869

Configure Domain Controllers for Smart Card Logon.

Rationale:

This is a DoD requirement; DoD Instruction 8520.2 [R1206] and DoD Directive 8190.3 [R1297] refer, and Joint Task Force-Global Network Operations (JTF-GNO) Communications Tasking Order (CTO 06-02) specifically directs implementation of Smart Card Logon (SCL) on all **NIPRNet** networks.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / Smart Card Logon
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / Smart Card Logon
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / Smart Card Logon

Evaluation Criteria:

1) Test:

Is the Domain Controller configured for SCL?

Procedure:

Verify that the Domain Controller is configured for SCL.

Example:

None.

G1883

Use a DoD PKI code signing certificate to sign mobile code residing on DoD-owned or DoD-controlled servers.

Rationale:

DoD Instruction 8552.01 [R1292] requires providing a DoD PKI issued code-signing certificate for all DoD-owned or DoD controlled servers. DoD code-signing certificates must be used to sign mobile code that will reside on DoD servers whenever possible.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Mobile Code

Evaluation Criteria:

1) Test:

Is mobile code residing on a DoD-owned or DoD-controlled server signed by a DoD code signing certificate from an approved DoD PKI Certificate Authority?

Procedure:

Verify that the mobile code has been signed.
Verify that the certificate was issued by a DoD PKI Certificate Authority that issues code signing certificates.

Example:

For signing mobile code using Mozilla/Netscape ***SignTool***:

- How to Sign Applets Using RSA-Signed Certificates: http://java.sun.com/j2se/1.4.2/docs/guide/plugin/developer_guide/rsa_signing.html
- Netscape Certificate Management System Administrator's Guide, Appendix F: http://docs.sun.com/source/816-5531-10/app_sign.htm
- Code Signing Digital IDs for Netscape Object Signing: http://www.verisign.com/resources/gd/objectSigning/index.html

For signing Java applets using Java ***Keytool***:

- How to Sign Applets Using RSA-Signed Certificates: http://java.sun.com/j2se/1.4.2/docs/guide/plugin/developer_guide/rsa_signing.html
- Keytool - Key and Certificate Management Tool: http://java.sun.com/j2se/1.3/docs/tooldocs/win32/keytool.html
- Code Signing Digital IDs for Sun Java Signing: http://www.verisign.com/resources/gd/javaSigning/index.html

For signing Microsoft Office ***VBA macros***:

- Code Signing Digital IDs for Microsoft Office 2000/Visual Basic for Applications: http://www.verisign.com/resources/gd/msOffice/index.html

For signing mobile code using Microsoft ***Signcode***:

- Signing and Checking Code With Authenticode: http://msdn.microsoft.com/workshop/security/authcode/signing.asp
- Code Signing Digital IDs for Microsoft Authenticode Technology: http://www.verisign.com/resources/gd/authenticode/index.html

For signing mobile code with Internet Explorer ***Administration Kit 5.0*** or later:

- Code Signing With IEAK 5 and Later: http://support.microsoft.com/default.aspx?scid=kb;en-us;269395

G1884

Configure browsers to use Category 1A allowed mobile code per DoD Instruction 8552.01. [R1292]

Rationale:

Required by DoD Instruction 8552.01 [R1292] to only allow ActiveX and Shockwave movies in browsers.

Note: *Microsoft Internet Explorer version 6/SP2 or version 7 is the only browser that is capable of executing ActiveX controls in compliance with the Category 1 usage restrictions.*

Note: *The lack of mobile code in a system does not constitute a waiver for the system.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Mobile Code

Evaluation Criteria:

1) Test:

Is the browser properly configured to comply with the Category 1A usage restrictions for ActiveX and Shockwave controls?

Procedure:

Verify configuration of the browser to comply with Category 1A usage restrictions for ActiveX and Shockwave.

Example:

G1885

Configure browsers to disable Category 1X prohibited mobile code per DoD Instruction 8552.01. [R1292]

Rationale:

Required by DoD Instruction 8552.01 [R1292] to disable the following prohibited Category 1X mobile code in browsers:

Mobile code scripts that execute in Windows Scripting Host or WSH (e.g., JavaScript and VBScript downloaded via a **Uniform Resource Locator** [**URL**] file reference or email attachment)

- HTML Applications (e.g., **`.HTA`** files) that download as mobile code
- Scrap objects
- Microsoft Disk Operating System (MS-DOS) batch scripts
- Unix shell scripts
- Binary executables (e.g., **`.exe`** files) that download as mobile code

Note: *The lack of mobile code in a system does not constitute a waiver for the system.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Mobile Code

Evaluation Criteria:

1) Test:

Is the browser properly configured to disable Category 1X prohibited mobile code?

Procedure:

Verify all Category 1X prohibited mobile code is disabled in the browser.

Example:

None.

G1886

Disable automatic execution of mobile code in email clients.

Rationale:

Due to the significant risk of malicious mobile code downloading into user workstations via email, and the ease of rapidly spreading malicious mobile code via email, the following restrictions apply to all types of mobile code in email independent of risk category:

- Disable the automatic execution of all categories of mobile code in email bodies and attachments .
- Configure desktop software to prompt the user prior to opening email attachments that may contain mobile code.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Mobile Code

Evaluation Criteria:

1) Test:

Is automatic execution of mobile code in email bodies and attachments disabled?

Procedure:

Verify that Category 1X mobile code file types have been disassociated.
Verify that execution of mobile code is disabled in an email body

Verify that execution of mobile code is disabled in an email attachment.

Example:

Some email client products, such as Microsoft Outlook and Outlook Express, use the Windows file type associations to select the appropriate application to process a file. Disassociating these file types in Windows will prevent the contents of files with those related file extensions from automatically executing whenever the user selects the file.

2) Test:

Is the user prompted prior to opening email attachments?

Procedure:

Verify that the user is prompted prior to opening email attachments containing mobile code.

Example:

DoD mobile code policy requires prompting the user prior to opening email attachments that may contain mobile code. Microsoft Outlook Express and Outlook use the Windows file types and settings. SeaMonkey and Thunderbird maintain their own internal file type settings. Windows should be configured to prompt users prior to opening downloaded files. In addition, Windows must be configured to always display all files and file extensions to enable users to determine the type of file they may be opening.

G1887

Monitor configured mobile code-enabled software to ensure it is in compliance with DoD Instruction 8552.01. [R1292]

Rationale:

The primary foundation for implementing the DoD Mobile Code Policy and protecting against malicious mobile code is the proper secure configuration of users' desktop workstation software. The policy requires immediate correction of all identified misconfigurations.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Mobile Code

Evaluation Criteria:

1) Test:

Is there a plan or process in place to configure mobile code properly on DoD systems?

Procedure:

Verify configuration of workstation and server mobile code-enabled software to be compliant with DoD Instruction 8552.01. [R1292]
Verify that all identified misconfigurations are corrected immediately.

Example:

G1895

Encrypt all Unclassified DoD Data at Rest (DAR) not releasable to the public stored on mobile computing devices.

Rationale:

DoD mandates encryption not only for Personally Identifiable Information (PII), but for all non-publicly released Unclassified information that is contained on mobile computing devices and removable media.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Data at Rest

Evaluation Criteria:

1) Test:

Is all non-publicly released Unclassified information contained on mobile computing devices and removable storage media encrypted?

Procedure:

Verify that a data at rest encryption product is properly installed and configured to encrypt DAR on mobile computing devices and removable storage media containing non-publicly released Unclassified information.

Example:

None.

G1896

Use Data at Rest (DAR) products that are Federal Information Processing Standard (FIPS) 140-2 compliant.

Rationale:

The Office of Management and Budget (OMB) and **DoD** require that all encryption products meet National Institute of Standards and Technology (**NIST**) **Federal Information Processing Standard** (**FIPS**) 140-2 requirements or have a National Security Agency (NSA) approval letter for use in U.S. Government networks.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Data at Rest

Evaluation Criteria:

1) Test:

Is the DAR encryption FIPS 140-2 compliant?

Procedure:

Verify that NIST has validated the cryptographic module to meet NIST FIPS 140-2 requirements or NSA has approved the module for use on government networks.

Example:

None.

G1897

Purchase Data at Rest (DAR) encryption products that are included in the Enterprise Software Initiative (ESI).

Rationale:

DoD components must purchase **data at rest** (**DAR**) encryption products to protect DAR on mobile computing devices and removable storage media through the Enterprise Software Initiative (ESI) since it benefits all of the DoD. All ESI awarded products are Federal Information Processing Standard (**FIPS**) 140-2 compliant, support Common Access Card (**CAC**)integration, licenses are transferable within a federal agency, and licenses include secondary use rights.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Data at Rest

Evaluation Criteria:

1) Test:

Are DAR encryption products purchased through the ESI?

Procedure:

Verify that DAR encryption products are purchased through the ESI.

Example:

None.

G1902

Use the Exclusive Canonicalization algorithm when digitally signing XML content that may be embedded in another XML document.

Rationale:

Namespaces are inherited from parent XML nodes. The digital signature of a signed XML message fragment from a source XML document placed into destination XML document may fail signature verification due to the inheritance of namespaces from the destination document. Exclusive canonicalization handles namespaces of surrounding XML content differently to support this use case.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures

Evaluation Criteria:

1) Test:

Are XML message fragments intented for inclusion in other XML documents canonicalized using the Exclusive Canonicalization algorithm?

Procedure:

Verify message fragments intented for inclusion in other XML documents are canonicalized using the Exclusive Canonicalization algorithm by inspecting the canonicalization method in the source code or the resulting signed XML **`CanonicalizationMethod`** element.

Example:

None.

G1910

Provide for transformation of XML messages using eXtensible Style Language Transformations (XSLT) when implementing an Enterprise Service Bus (ESB).

Rationale:

Mediation, including transformation, is a core characteristic of an ESB. XSLT is the most commonly used standards-based language for transforming XML.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Enterprise Service Bus (ESB)
NESI / Part 5: Developer Guidance / Middleware / Enterprise Service Bus (ESB)

Evaluation Criteria:

1) Test:

Does the implemented ESB architecture provide for transformation of XML messages using XSLT?

Procedure:

Verify that the implemented ESB architecture provide for transformation of XML messages using XSLT?

Example:

G1912

Support the execution of Business Process Execution Language (BPEL) when implementing an Enterprise Service Bus (ESB).

Rationale:

BPEL is an orchestration language for executing business process through the arrangement, coordination and management of services into composite services. BPEL orchestration capabilities within an ESB allows for a standards-based way to execute buisness processes. Business process orchestration outside of an ESB may lead to duplicative and conflicting mediation and registration capabilities.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Enterprise Service Bus (ESB)
NESI / Part 5: Developer Guidance / Middleware / Enterprise Service Bus (ESB)

Evaluation Criteria:

1) Test:

Does the ESB provide support for the execution of BPEL?

Procedure:

Verify that the ESB is able to execute BPEL.

Example:

BP1007

Develop software using open standard Application Programming Interfaces (APIs).

Rationale:

Using open standard APIs enables code portability and reduces dependancies on proprietary APIs.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the application create customized/proprietary solutions where standardized **API**s exists?

Procedure:

Check the application for code that has proprietary solutions where standardized APIs exists.

Example:

None

BP1021

Create fully encapsulated classes.

Rationale:

Data members should not be public as making implementation details public creates interdependencies between the class and its users, subjecting the users to changes in implementation. Therefore, access should only occur via public interface methods. This makes the implementation more robust, because all data can be validated when assigned new values and allows for logging of changed values.

Referenced By:

NESI / Part 2: Traceability / Naval Open Architecture / Maintainability
NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Do instance variables have public access or are they more accessible than necessary?

Procedure:

Check that the instance variable in classes does not have public access unless it is static and final.

Example:

None

2) Test:

Does the class provide direct access to internal data via pass by reference?

Procedure:

Check to make sure that the methods that access the internal state do not return a reference to the internal data.

Example:

None

BP1038

Use a sans serif font (e.g., Arial, Verdana) in Web pages rather than a serif font (e.g., Times New Roman).

Rationale:

Web pages are easier to read with **sans serif** fonts.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Style Sheets

BP1039

Do not underline any text unless it is a link.

Rationale:

Underlined text is the default behavior of an **HTML** link. Many users consider this the norm and may find a **Web page** difficult to read if other items are underlined.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces

BP1040

Use hex codes for all colors (e.g., #FFFF33), never the color name (e.g., yellow).

Rationale:

Using hex codes for colors is a common industry practice to increase compatibility between browsers.

For an online hexadecimal color chart, see http://webmonkey.wired.com/webmonkey/reference/color_codes/.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Style Sheets

BP1041

Do not change the default colors of the links.

Rationale:

Web pages are easier to read because users have become accustomed to the default colors.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Style Sheets
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Style Sheets

BP1042

Do not build a Web page where the horizontal width is greater than the screen (vertical scrolling is fine), planning for the lowest common denominator to be super-VGA resolution (800 x 600).

Rationale:

This enables a user to print pages on most printers and render pages on most displays.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces

BP1054

Use standard controls that provide input choices for the user.

Rationale:

Using standard controls such as radio buttons, check boxes, list boxes, and drop-downs reduces user input errors and aids in data integrity.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

BP1097

Use the `System.Text.StringBuilder` class for repetitive string modifications such as appending, removing, replacing, or inserting characters.

Rationale:

Strings in **.NET** are immutable. This means that every time a string is created as a result of a string operation such as concatenation, a new string is created for each intermediate string in a set of operations. This has a lot of string management overhead. **`StringBuilder`** avoids these problems.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / .NET Framework
NESI / Part 5: Developer Guidance / Middleware / .NET Framework

Evaluation Criteria:

1) Test:

Are there repetitive string operations that use string operations instead of **`StringBuilder`** operations?

Procedure:

Scan all C# code for repetitive string operations such as appending, removing, replacing, or inserting characters.

Example:

None

BP1098

Write all .NET code in C#.

Rationale:

Because of the high degree of similarities between C# and Java, .NET code written in C# is easily ported to Java. .NET has removed most of the advantages of one language (C#, C++, J++, VB) over another.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / .NET Framework
NESI / Part 5: Developer Guidance / Middleware / .NET Framework

Evaluation Criteria:

1) Test:

Are any .NET languages delivered other than C#?

Procedure:

Scan delivered code for registered .NET file extensions other than C#.

Example:

None

BP1100

Compile all .NET code using the .NET Just-In-Time compiler.

Rationale:

There are two different ways to generate machine code within the .NET environment: **Just-In-Time (JIT)** and Native Image Generator (**NGEN**). The NGEN method provides performance advantages by using the native image cache portion of the global assembly cache, which is specific to the machine where the .NET **common language runtime** is installed. It is machine-dependent and is less portable.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / .NET Framework
NESI / Part 5: Developer Guidance / Middleware / .NET Framework

Evaluation Criteria:

1) Test:

Is **`ngen.exe`** used?

Procedure:

Scan all delivered code for the use of **`ngen.exe`** or the **ngen** command.

Example:

None

BP1111

Mark all Microsoft Message Queue (MSMQ) messages as recoverable.

Rationale:

MSMQ normally only stores the contents of messages in memory, which will be lost if a power, hardware, or software failure occurs. By marking messages as recoverable, messages are also stored to disk so the contents can be recovered after a failure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Messaging with MSMQ
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Messaging with MSMQ
NESI / Part 5: Developer Guidance / Middleware / Messaging / Messaging with MSMQ

Evaluation Criteria:

1) Test:

Are all messages and message queues marked as recoverable?

Procedure:

Scan the code for the creation of messages and message codes, and make sure each has the **`recoverable`** attribute set to true.

Example:

None

BP1112

Specify all Microsoft Message Queue (MSMQ) queues as transactional if they support multiple-step processes.

Rationale:

Transactions allow multi-step processes to behave correctly when a **rollback** occurs.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Messaging with MSMQ
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Messaging with MSMQ
NESI / Part 5: Developer Guidance / Middleware / Messaging / Messaging with MSMQ

BP1116

If using Java-based messaging (e.g., JMS), register destinations in Java Naming and Directory Interface (JNDI) so message clients can use JNDI to look up these destinations.

Rationale:

JNDI is an industry standard for Java-based applications. Many JMS interoperability coding issues relate to the publication and discovery of JNDI for resources. To mitigate these issues, encapsulate resource definitions in a properties file or in **Java EE** as a **deployment descriptor**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / JNDI Security

BP1139

Do not use proprietary SQL extensions.

Rationale:

The use of proprietary extensions increases vendor dependence.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Have the developers adhered to a core set of features and minimized use of proprietary extensions to the **SQL** standard?

Procedure:

Examine a representative sample of database scripts and stored procedures.

Example:

None

BP1140

Use SQL-2003 features in preference to SQL-92 or SQL-99.

Rationale:

SQL-2003 includes many **XML** and **OODB** extensions and features. Use it in preference to SQL-99 or SQL-92 entry-level features to justify the recommendations against using native XML databases and OODB databases.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Have the developers used SQL-2003 features rather than SQL-92 or SQL-99 features?

Procedure:

Examine a representative sample of database scripts and stored procedures.

Example:

None

BP1143

Use a database modeling tool that supports a two-level model (Conceptual/Logical and Physical) and ISO-11179 data exchange standards.

Rationale:

ISO-11179 is a **metadata** repository standard. Supporting tools store the model locally in an **XML** file or in a vendor-specific repository. For many applications, there is no need to use the repository at all. **Configuration Management** could be affected by checking the model in and out of a tool such as Source Safe. Entity-Relationship **data model** is synonymous with a **Conceptual data model**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Is a database modeling tool being used and does it support the **ISO-11179** data exchange standards?

Procedure:

Verify that the requirement for a database modeling tool is included in the system requirements. If ISO-11179 standard-based repository products become available, determine whether the product provides an interface thereto.

Example:

None

BP1145

Use vendor-neutral conceptual/logical models.

Rationale:

The leading database vendors do not have a common set of data types or object name length limitations, and there are no **ANSI** standards that address these issues. To maintain vendor-neutral models, do not accept vendor-specific features.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

Has the data model been designed using vendor-neutral design criteria?

Procedure:

Examine the conceptual/logical data model.

Example:

None

BP1227

Do not allow installation of MSMQ-dependent clients.

Rationale:

MSMQ-dependent clients require synchronous access to an MSMQ server and create performance issues on the server. Consequently, dependent clients cannot operate if they are disconnected from the rest of the **enterprise** networks.

Dependent clients cannot be run under local accounts.

Dependent clients leave all encrypted messages in plain text between the client and server.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Messaging with MSMQ
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Messaging with MSMQ
NESI / Part 5: Developer Guidance / Middleware / Messaging / Messaging with MSMQ
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1230

Do not use the MSMQ `SupportLocalAccountsOrNT4` feature.

Rationale:

This entry enables weakened security for Active Directory on a **domain** controller, which is then replicated to all other domain controllers in every domain in your forest.

See the Microsoft Message Queuing Web site for additional information.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Messaging with MSMQ
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Messaging with MSMQ
NESI / Part 5: Developer Guidance / Middleware / Messaging / Messaging with MSMQ

BP1231

Use `CORBA::String_var` in IDL to pass string types in C++.

Rationale:

Follow this practice to correct memory management and reduce memory leaks and runtime faults.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA

Evaluation Criteria:

1) Test:

Is **`String_var`** used in the implementation code that was not auto generated?

Procedure:

Check implementation code that was not autogenerated for all occurrences of "string" and verify that they are **`String_var`** .

Example:

None

BP1232

Do not pass or return a zero or null pointer; instead, pass an empty string.

Rationale:

Follow this practice to correct memory management and reduce memory leaks and runtime faults.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA

Evaluation Criteria:

1) Test:

Are there any returns that contain pointers that are assigned zero?

Procedure:

Check code to make sure that all strings returned always have a safety check for zero or null pointers, and assign them to empty strings.

Example:

None

BP1233

Do not assign `CORBA::String_var` type to `INOUT` method parameters.

Rationale:

Follow this practice to correct memory management and reduce memory leaks and runtime faults.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA

Evaluation Criteria:

1) Test:

Are there any implementation classes using methods that contain **`CORBA::String_var`**?

Procedure:

Inspect CORBA code to make sure **`INOUT`** parameters are not assigned to **`CORBA::String_var`** values.

Example:

None

BP1234

Assign string values to `OUT` , `INOUT` , or `RETURN` parameters using operations to allocate or duplicate values rather than creating and deleting values.

Rationale:

Correct memory management and reduce memory leaks and reduce runtime faults.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA

Evaluation Criteria:

1) Test:

Are **`string_dup`**, **`string_alloc`** and **`string_free`** being used?

Procedure:

Search CORBA code for the use of **`string_dup`**, **`string_alloc`**, and **`string_free`**.

Example:

None

2) Test:

Are new and delete operators being used for strings being assigned to **`OUT`**, **`INOUT`**, or **`RETURN`** parameters?

Procedure:

Inspect CORBA code to make sure **`OUT`**, **`INOUT`**, and **`RETURN`** parameters are not using strings managed with the new and delete operators.

Example:

None

BP1235

Assign string values to returned-as-attribute values using operations to allocate or duplicate values rather than creating and deleting values.

Rationale:

Follow this practice to correct memory management and reduce memory leaks and runtime faults.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / CORBA
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / CORBA
NESI / Part 5: Developer Guidance / Middleware / CORBA

Evaluation Criteria:

1) Test:

Are **`string_dup`**, **`string_alloc`**, and **`string_free`** being used?

Procedure:

Search CORBA code for the use of **`string_dup`**, **`string_alloc`**, and **`string_free`**.

Example:

None

2) Test:

Are new and delete operators being used for strings being returned-as-attribute?

Procedure:

Inspect CORBA code to make sure returned-as-attribute string values are not using strings managed with the new and delete operators.

Example:

None

BP1240

Present complete and coherent sets of concepts to the user.

Rationale:

The **interface** should not require the consumer continually to implement multiple interfaces when a single interface can accomplish the same thing.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design

BP1241

Design statically typed interfaces.

Rationale:

Designing a statically typed interface allows consumers to use early binding rather than late binding. This minimizes the risk for runtime errors due to late binding.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design

BP1242

Minimize an interface's dependencies on other interfaces.

Rationale:

Minimizing the dependency of an interface on other interfaces simplifies the use of the interface by consumers.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design

BP1243

Express interfaces in terms of application-level types.

Rationale:

Use application-level types to maintain the meaning of values used with the interface. This enables data validation and other runtime safety checks against the data.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design

BP1244

Use assertions only to aid development and integration.

Rationale:

Assertions allow evaluating Boolean expressions to determine if the code is executing within the proper operating constraints. For example, if a calculated temperature is supposed to be between -273 degrees and +1,000 degrees, it is possible to test the results of the calculation with an assertion. Once the code is tested and/or integrated, this calculation no longer needs to occur after each calculation.

Assertion execution is integrated into the **compiler**. Consequently, it is possible to add it into the executable or eliminate it by setting compiler options (i.e., switches). Assertions are therefore ideal for adding code that is useful during development or integration, but wasteful in delivered code.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / C4ISR: Payload Platform / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Public Interface Design
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Public Interface Design
NESI / Part 5: Developer Guidance / Public Interface Design

Evaluation Criteria:

1) Test:

Do public methods that implement interfaces have assertions?

Procedure:

Check all implementations of public interfaces to ensure that all public methods that are part of the interface do not use the **`assert`** command.

Example:

The following example shows a correct implementation of a public method in a public interface.

```
public interface NameInterface is
public String getName
  ( int nameID )
  Throws IllegalArgumentException
  {
    /* precondition check */
    if ( nameID <= 0
         || nameID > MAX_NAMES
       )
    { throw new IllegalArgumentException
        ("Illegal id number: " + nameID);
    }
    . . .// Do the computation
    return theResult;
  } // End getName
} // NameInterface
```

The following example shows an incorrect implementation of a public method in a public interface. Do not use the implementation exemplified by the red code.

```
public interface NameInterface is
public String getName
  ( int nameID )
  {
    /* precondition check */
    assert nameID <= 0
         || nameID > MAX_NAMES
       : "Illegal id number: " + nameID);
```

```
... . . .// Do the computation
    return theResult;
  } // End getName
} // NameInterface
```

BP1246

Base Java-based portlets on JSR 168.

Rationale:

JSR 168 enables **interoperability** between Java **portlets** and **portals**. This specification defines a set of **APIs** for portal computing that addresses the areas of aggregation, **personalization**, presentation, and security. http://www.jcp.org/en/jsr/detail?id=168

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Web Portals
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Web Portals

BP1247

Encapsulate Java-based portlets in a .war file.

Rationale:

Storing JSR-168-compliant code in the portal container improves **interoperability** and code reuse.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients / Web Portals
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients / Web Portals

BP1248

Follow a naming convention.

Rationale:

The names of schemas, users, tables, and columns need to be unique and descriptive. Unfortunately, it is possible (but undesirable) to give the same name to multiple objects; for example, assigning the name "employee" to a database, table, and column. Many naming conventions get around this by appending a suffix that indicates the kind of object: for example, **`Employee_Db`**, **`Employee_Tbl`**, **`Employee_Id`**, **`Employee_Indx`**.

Avoid generic column names such as "ID." Systems often have many kinds of IDs, and even if the system really only does have a single ID, it will be more difficult to merge with other databases if they have also used the column name "ID."

Some DBMSs support mixed-case names of unlimited length, while others are case-insensitive. For portability, assume that names are case-insensitive and limited to 30 characters. Do not use reserved words from the **SQL-92**, **SQL:1999**, or SQL:2003 standards.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Is there a naming convention?

Procedure:

Check for the existence of a document that governs naming conventions, or look for patterns in the database metadata.

Example:

Use database commands to look at the database metadata:

```
select username from all_users
select table_name from user_tables
select index_name from user_indexes
```

BP1249

Do not use generic names for database objects such as databases, schema, users, tables, views, or indices.

Rationale:

Assigning generic names to user-defined objects within a database can lead to confusion and unexpected results. For example, naming a database "instance" within the **RDBMS** database is confusing to the humans who have to read commands that reference the database. In addition, the RDBMS software may parse it incorrectly.

> ***Note:*** *Although some RDBMS interpreters allow the use of a generic or reserved word to name objects if the name is surrounded with quotes, this is not a recommended practice.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Are any generic names used for user-defined objects?

Procedure:

Examine the RDBMS metadata for generic names such as database, table, entity, column, attribute, select, view, etc.

Example:

```
select table_name from user_tables where table_name in ('database','entity',...)
select column_name from user_tab_columns where column_name in ('database','entity',...)
```

BP1250

Use case-insensitive names for database objects such as databases, schema, users, tables, views, and indices.

Rationale:

The **SQL** standard does not require names to be case-sensitive. Consequently, some DBMSs are not case-sensitive. Using case-sensitive names, therefore, makes portability more difficult.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Are the names of database objects case-sensitive?

Procedure:

Examine the database metadata for "run-on" names. If the database supports case-sensitive names, check to see if it is using camel-back capitalization.

Example:

```
EMPLOYEEBENEFITSTBL
EmployeeBenefitsTbl
```

BP1251

Separate words with underscores.

Rationale:

The **SQL** standard does not require names to be case-sensitive. Consequently, some DBMSs are not case-sensitive. Using case-sensitive names, therefore, makes portability more difficult. To avoid these problems, use underscores to separate words (**`employee_benefits_tbl`**) rather than camel-back capitalization (**`EmployeeBenefitsTbl`**).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Are underscores used between the words in the names of database objects?

Procedure:

Examine the database metadata and look for names that do not have underscores separating words.

Example:

```
EMPLOYEEBENEFITSTBL versus
EMPLOYEE_BENEFITS_TBL
EmployeeBenefitsTbl versus
Employee_Benefits_Tbl
```

BP1252

Do not use names with more than 30 characters.

Rationale:

Not all DBMSs support unlimited name lengths. For example, Oracle limits object names to 30 characters. Therefore, using names longer than 30 characters can reduce portability by limiting the DBMSs on which the system can be deployed.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Are any of the database object names more than 30 characters in length?

Procedure:

Examine the database metadata and look for names that are longer than 30 characters.

Example:

```
W2_EMPLOYEE_BENEFITS_FOR_FAMILIES_TBL
```

BP1253

Do not use the SQL:1999 or SQL:2003 reserved words as names for database objects such as databases, schema, users, tables, views, or indices.

Rationale:

Using reserved words as the names of database objects can cause ambiguities and errors. It limits the ability to upgrade or port the code to other systems.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Are any of the **SQL:1999** or SQL:2003 reserved words used to name objects in the database?

Procedure:

Examine the database metadata for names that are in the list of SQL:1999 or SQL:2003 reserved words

Example:

Look for any of these words:

```
ABS ABSOLUTE ACCESS ACQUIRE ACTION ADA ADD ADMIN AFTER AGGREGATE ALIAS ALL ALLOCATE ALLOW ALTER AND ANY ARE
ARRAY AS ASC ASENSITIVE ASSERTION ASUTIME ASYMMETRIC AT ATOMIC AUDIT AUTHORIZATION AUX AUXILIARY AVG
BACKUP BEFORE BEGIN BETWEEN BIGINT BINARY BIT BIT_LENGTH BLOB BOOLEAN BOTH BREADTH BREAK BROWSE BUFFERPOOL
BULK BY
CALL CALLED CAPTURE CARDINALITY CASCADE CASCADED CASE CAST CATALOG CCSID CEIL CEILING CHAR CHAR_LENGTH
CHARACTER CHARACTER_LENGTH CHECK CHECKPOINT CLASS CLOB CLOSE CLUSTER CLUSTERED COALESCE COLLATE COLLATION
COLLECT COLLECTION COLLID COLUMN COMMENT COMMIT COMPLETION COMPRESS COMPUTE CONCAT CONDITION CONNECT
CONNECTION CONSTRAINT CONSTRAINTS CONSTRUCTOR CONTAINS CONTAINSTABLE CONTINUE CONVERT CORR CORRESPONDING
COUNT COUNT_BIG COVAR_POP COVAR_SAMP CREATE CROSS CUBE CUME_DIST CURRENT CURRENT_COLLATION CURRENT_DATE
CURRENT_DEFAULT_TRANSFORM_GROUP CURRENT_LC_PATH CURRENT_PATH CURRENT_ROLE CURRENT_SERVER CURRENT_TIME
CURRENT_TIMESTAMP CURRENT_TIMEZONE CURRENT_TRANSFORM_GROUP_FOR_TYPE CURRENT_USER CURSOR CYCLE
DATA DATABASE DATALINK DATE DAY DAYS DB2GENERAL DB2SQL DBA DBCC DBINFO DBSPACE DEALLOCATE DEC DECIMAL DECLARE
DEFAULT DEFERRABLE DEFERRED DELETE DENSE_RANK DENY DEPTH DEREF DESC DESCRIBE DESCRIPTOR DESTROY DESTRUCTOR
DETERMINISTIC DIAGNOSTICS DICTIONARY DISALLOW DISCONNECT DISK DISTINCT DISTRIBUTED DLNEWCOPY DLPREVIOUSCOPY
DLURLCOMPLETE DLURLCOMPLETEONLY DLURLCOMPLETEWRITE DLURLPATH DLURLPATHONLY DLURLPATHWRITE DLURLSCHEME
DLURLSERVER DLVALUE DO DOMAIN DOUBLE DROP DSSIZE DUMMY DUMP DYNAMIC
EACH EDITPROC ELEMENT ELSE ELSEIF END END-EXEC EQUALS ERASE ERRLVL ESCAPE EVERY EXCEPT EXCEPTION EXCLUSIVE
EXEC EXECUTE EXISTS EXIT EXP EXPLAIN EXTERNAL EXTRACT
FALSE FENCED FETCH FIELDPROC FILE FILLFACTOR FILTER FINAL FIRST FLOAT FLOOR FOR FOREIGN FORTRAN FOUND FREE
FREETEXT FREETEXTTABLE FROM FULL FUNCTION FUSION
GENERAL GENERATED GET GLOBAL GO GOTO GRANT GRAPHIC GROUP GROUPING
HANDLER HAVING HOLD HOLDLOCK HOST HOUR HOURS
IDENTIFIED IDENTITY IDENTITY_INSERT IDENTITYCOL IF IGNORE IMMEDIATE IMPORT IN INCLUDE INCREMENT INDEX
INDICATOR INITIAL INITIALIZE INITIALLY INNER INOUT INPUT INSENSITIVE INSERT INT INTEGER INTEGRITY INTERSECT
INTERSECTION INTERVAL INTO IS ISOBID ISOLATION ITERATE
JAR JAVA JOIN
KEY KILL
LABEL LANGUAGE LARGE LAST LATERAL LC_CTYPE LEADING LEAVE LEFT LESS LEVEL LIKE LIMIT LINENO LINKTYPE LN LOAD
LOCAL LOCALE LOCALTIME LOCALTIMESTAMP LOCATOR LOCATORS LOCK LOCKSIZE LONG LOOP LOWER
MAP MATCH MAX MAXEXTENTS MEMBER MERGE METHOD MICROSECOND MICROSECONDS MIN MINUS MINUTE MINUTES MOD MODE
MODIFIES MODIFY MODULE MONTH MONTHS MULTISET
NAME NAMED NAMES NATIONAL NATURAL NCHAR NCLOB NEW NEXT NHEADER NO NOAUDIT NOCHECK NOCOMPRESS NODENAME
NODENUMBER NONCLUSTERED NONE NORMALIZE NOT NOWAIT NULL NULLIF NULLS NUMBER NUMERIC NUMPARTS
```

```
OBID OBJECT OCTET_LENGTH OF OFF OFFLINE OFFSETS OLD ON ONLINE ONLY OPEN OPENDATASOURCE OPENQUERY OPENROWSET
OPENXML OPERATION OPTIMIZATION OPTIMIZE OPTION OR ORDER ORDINARILITY OUT OUTER OUTPUT OVER OVERLAPS OVERLAY
PACKAGE PAD PAGE PAGES PARAMETER PARAMETERS PART PARTIAL PARTITION PASCAL PATH PCTFREE PCTINDEX PERCENT
PERCENT_RANK PERCENTILE_CONT PERCENTILE_DISC PIECESIZE PLAN POSITION POSTFIX POWER PRECISION PREFIX PREORDER
PREPARE PRESERVE PRIMARY PRINT PRIOR PRIQTY PRIVATE PRIVILEGES PROC PROCEDURE PROGRAM PSID PUBLIC
QUERYNO
RAISERROR RANGE RANK RAW READ READS READTEXT REAL RECONFIGURE RECOVERY RECURSIVE REF REFERENCES REFERENCING
REGR_AVGX REGR_AVGY REGR_COUNT REGR_INTERCEPT REGR_R2 REGR_SLOPE REGR_SXX REGR_SXY REGR_SYY RELATIVE RELEASE
RENAME REPEAT REPLICATION RESET RESIGNAL RESOURCE RESTORE RESTRICT RESULT RETURN RETURNS REVOKE RIGHT ROLE
ROLLBACK ROLLUP ROUTINE ROW ROW_NUMBER ROWCOUNT ROWGUIDCOL ROWID ROWNUM ROWS RRN RULE RUN
SAVE SAVEPOINT SCHEDULE SCHEMA SCOPE SCRATCHPAD SCROLL SEARCH SECOND SECONDS SECQTY SECTION SECURITY SELECT
SENSITIVE SEQUENCE SESSION SESSION_USER SET SETS SETUSER SHARE SHUTDOWN SIGNAL SIMILAR SIMPLE SIZE SMALLINT
SOME SOURCE SPACE SPECIFIC SPECIFICTYPE SQL SQLCA SQLCODE SQLERROR SQLEXCEPTION SQLSTATE SQLWARNING SQRT
STANDARD START STATE STATEMENT STATIC STATISTICS STAY STDDEV_POP STDDEV_SAMP STOGROUP STORES STORPOOL
STRUCTURE STYLESUBPAGES SUBSTRING SUCCESSFUL SUM SYMMETRIC SYNONYM SYSDATE SYSTEM SYSTEM_USER
TABLE TABLESPACE TEMPORARY TERMINATE TEXTSIZE THAN THEN TIME TIMESTAMP TIMEZONE_HOUR TIMEZONE_MINUTE TO TOP
TRAILING TRAN TRANSACTION TRANSLATE TRANSLATION TREAT TRIGGER TRIM TRUE TRUNCATE TSEQUAL TYPE
UID UNDER UNDO UNION UNIQUE UNKNOWN UNNEST UNTIL UPDATE UPDATETEXT UPPER USAGE USE USER USING
VALIDATE VALIDPROC VALUE VALUES VAR_POP VAR_SAMP VARCHAR VARCHAR2 VARIABLE VARIANT VARYING VCAT VIEW VOLUMES
WAITFOR WHEN WHENEVER WHERE WHILE WIDTH_BUCKET WINDOW WITH WITHIN WITHOUT WLM WORK WRITE WRITETEXT
YEAR YEARS
ZONE
```

BP1254

For command-and-control systems, use the names defined in the Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM) for data exposed to the outside communities.

Rationale:

The **Command-and-Control** (**C2**) **COI** has developed a **data model** to facilitate the exchange of data within the community and by consumers of their data outside the community. Therefore, data that is to be exposed from the database to the **COI** community or its data consumers should defer to the **data model** whenever possible. The JC3IEDM [R1070] data model defines the data units as well as the names and structure of the data.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

Evaluation Criteria:

1) Test:

If this is a system, does it use for the data that is exposed to the outside world?

Procedure:

Review all the data that is exposed to the outside world and confirm that it conforms to the JC3IEDM specifications.

Example:

None

BP1255

Use surrogate keys.

Rationale:

A surrogate key, also referred to as a system-generated key, database-sequence number, or arbitrary unique identifier, is a unique, arbitrary **primary key**. The **RDBMS** usually generates the surrogate key, but a database access layer such as the middle tier can also generate the surrogate key. The surrogate key is arbitrary because it is not derived from any data that exists within the table or the database. Another option for surrogate keys is Universally Unique Identifiers (UUIDs) (http://en.wikipedia.org/wiki/Universally_Unique_Identifier), the most common implementation being Microsoft's Globally Unique Identifiers (GUIDs) (http://en.wikipedia.org/wiki/Globally_Unique_Identifier).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1256

Use surrogate keys as the primary key.

Rationale:

Instead of using the natural keys to identify each record uniquely, use a surrogate key. This allows the natural key information to be modified independently of the primary key and any foreign-key references to the key.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

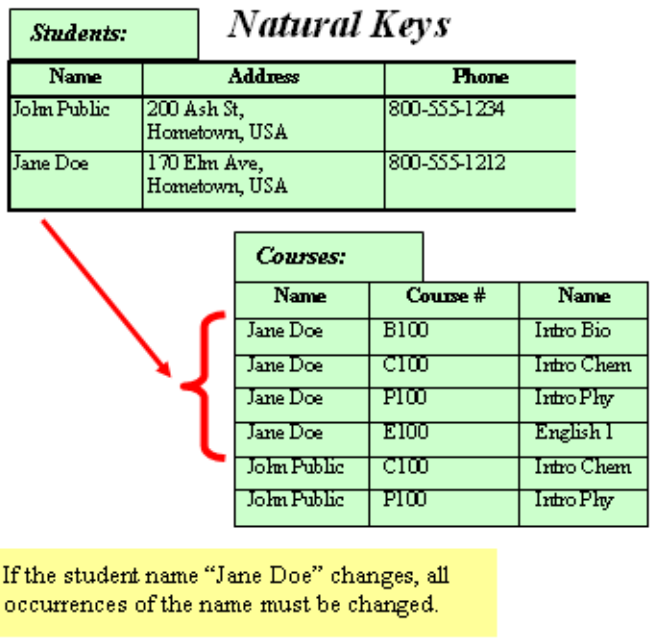
Are surrogate keys used instead of natural keys?

Procedure:

Look at the database metadata and determine if it uses surrogate or natural keys.

Example:

The following example shows natural keys. The primary keys are made up completely or in part from naturally occurring data in the tables.



If the student name "Jane Doe" changes, all occurrences of the name must be changed.

I1120

The following example shows a surrogate key being used instead of a natural key. Maintaining data is less complex than it is with natural keys and consequently less error-prone.

BP1257

Place a unique key constraint on the natural key fields.

Rationale:

Surrogate keys make it easier to maintain data. However, a column or set of columns should still uniquely identify the row in the table. This column or set of columns is the "natural key" or "secondary key." This natural key should still be protected by the uniqueness constraint normally associated with a **primary key**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Is there a unique key index for all tables that includes a column or set of columns not including the primary key?

Procedure:

Look at the database metadata to ensure that each table has a unique key, and that the columns in the unique key are not also part of the primary key.

Example:

BP1258

Explicitly define the encoding style of all data transferred via XML.

Rationale:

By default, **XML** is encoded using **Unicode**. Consequently, data transferred via XML should explicitly specify the encoding style. Assuming the default can cause **interoperability** problems between implementations.

Note: *Look for the following XML tag as the first line returned from queries that return XML from the database:*
`<?xml version="1.0" encoding="UTF-8"?>`

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Syntax
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Syntax
NESI / Part 5: Developer Guidance / Data / XML / XML Syntax

BP1259

Use indexes.

Rationale:

An index in an **RDBMS** is a summary of information organized to minimize the search time. Indexes summarize the information in a table. So, an employee table might have an index of last names, or last name and first name.

Having additional indexes on tables involves a tradeoff between query performance and insert/update/delete performance, which requires underlying index maintenance.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1260

Define a primary key for all tables.

Rationale:

By definition, a **primary key** uniquely defines each row within a table. To optimize the use of the table and to find records by the primary key, there should be an index that enforces the uniqueness of the key.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

Evaluation Criteria:

1) Test:

Is there a primary key defined for each table listed in the database?

Procedure:

Examine the database metadata to ensure there is a primary key for each table in the database.

Example:

BP1261

Monitor and tune indexes according to the response time during normal operations in the production environment.

Rationale:

Index efficiency depends on the data being indexed. Common variables follow:

- A sparsely populated table versus a densely populated table
- Data added in an presorted order versus a random order

Consequently, as the data changes, the efficiency of the index changes.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1262

In the case of Oracle, define indexes against the foreign keys (FK) columns to avoid contention and locking issues.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1263

Gather storage requirements in the planning phase, and then allocate twice the estimated storage space.

Rationale:

Storage space on the disk always poses a problem for databases, so it is necessary to plan storage space carefully.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1264

For high availability, use hardware solutions when geographic proximity permits.

Rationale:

There are many ways to achieve high availability. Some are based on hardware and others on software. As a general rule, hardware solutions use simple redundancy and are consequently less complex and fragile. If geographic proximity is not an issue, the hardware solution is preferable.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Relational Database Management Systems
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Relational Database Management Systems
NESI / Part 5: Developer Guidance / Data / Relational Database Management Systems

BP1265

Validate XML documents during document generation.

Rationale:

All **XML** passed between two systems or services must be valid. The XML document generator is responsible for ensuring that the document is valid and **well-formed**. If there are problems, the document generator is the only user that can effectively change the document.

Validity is checked via the use of a **W3C** Standard Validating parser. These parsers are built into most XML editors but are also available as stand alone products. Either the XML is valid or diagnostics are returned indicating where the XML is invalid.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XML Validation
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XML Validation
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XML Validation

Evaluation Criteria:

1) Test:

Are all the documents exported from the system or service valid and **well-formed**?

Procedure:

Capture all the documents and validate them, using an XML editor or stand alone XML validation tool.

Example:

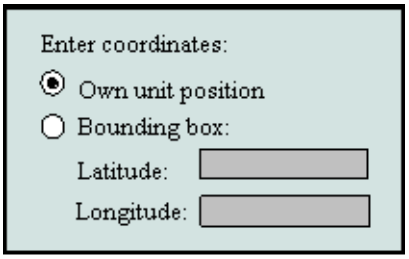
None.

BP1272

Disable dependent child controls when the parent control is inactive.

Rationale:

This practice makes it easier for the user to understand that the child controls depend on the selection of the parent, contributing to data integrity.



I1122

Referenced By:

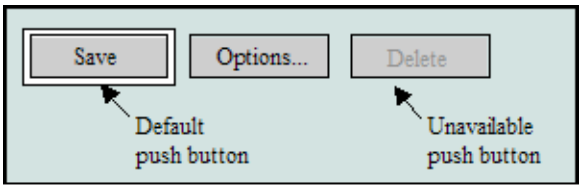
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

BP1273

Gray out the push button label if a button is unavailable.

Rationale:

This practice makes it easier for the user to understand that the button cannot be used until other action is taken.



I1126

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

BP1280

In tabular data displays, right justify integer data.

Rationale:

Whole numbers, displayed in a column, are easier to read if the digits of the same magnitude (1's, 10's, 100's, etc.) are vertically aligned.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

Evaluation Criteria:

1) Test:

Are all tabular whole number data right-justified?

Procedure:

Search all style sheets for the word "text-align." Examine the results for tabular whole number data and make sure the "text-align" attribute is set to "right"; visual Web page inspection may necessary to see if a defined align style is used within the tabular data.

Example:

Correct usage:

Cascading style sheet:

```
.td-items {
   text-align:right;
}
```

HTML:

Incorrect usage:

No alignment or incorrect alignment used.

BP1281

In tabular data displays, justify numeric data with decimals by using the decimal point.

Rationale:

It is common practice to align non-whole numbers by the decimal point for readability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

Evaluation Criteria:

1) Test:

Are all tabular non-whole number data justified by decimal point?

Procedure:

Search all style sheets for the word "text-align." Examine the results for tabular non-whole number data and make sure the "text-align" attribute is set to "."; visual Web page inspection may be necessary to see if a defined align style is used within the tabular data.

Example:

Correct usage:

Cascading style sheet:

```
.td-subtotal {
   text-align:".";
}
```

HTML:

Incorrect usage:

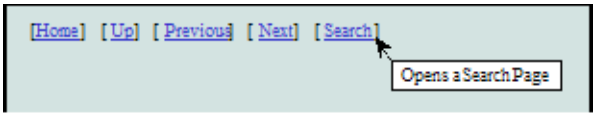
No alignment or incorrect alignment used.

BP1290

Use a tool tip to display help information about a control when the purpose of the control is not self-evident.

Rationale:

Using a tool tip increases user efficiency by preventing click errors. A mouse over event is the typical mapping for invoking a tool tip.



I1135

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

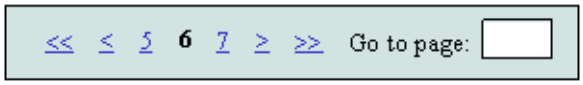
BP1291

Use obvious navigation controls for moving between pages in search results that span multiple pages.

Rationale:

Obvious navigation controls help a user to identify and use paging controls quickly. For example,

<	navigate back one page
>	navigate forward one page
<<	navigate back to the beginning page
>>	forward to the end page



I1136

Referenced By:

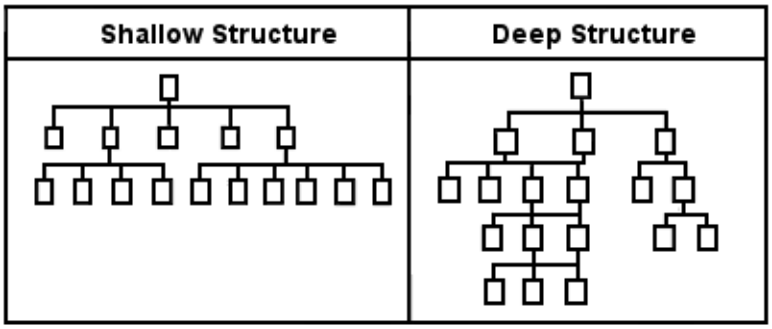
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients

BP1297

Structure a Web site hierarchy so users can reach important information and/or frequently accessed functions in a maximum of three jumps.

Rationale:

Use a shallow structure rather than a deep structure. A user's success at finding a target drops off sharply after three clicks.



I1139

Referenced By:

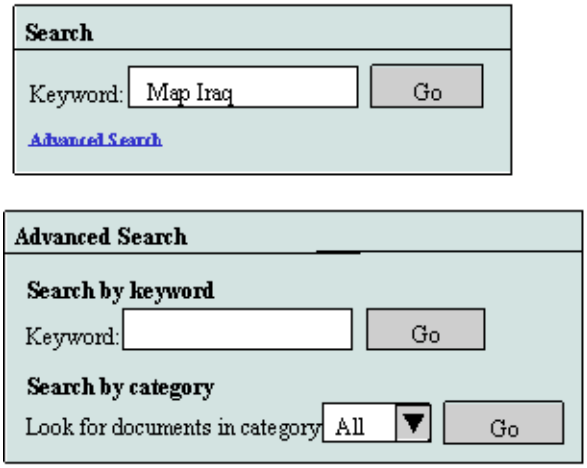
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces

BP1298

Provide basic search functionality as the default with a link or button that provides more advanced search features.

Rationale:

This practice makes the search feature cleaner and easier to use because the advanced features are hidden.



I1140

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

BP1299

Include a link back to the home page on all Web pages.

Rationale:

A link back to a Web site home page, for example in the form of a logo and a regular HTML link called **`Home`**, helps users navigate the Web site.

I1143

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces

BP1353

Use a data abstraction layer between the RDBMS and application for externally-visible applications to prevent the disclosure of sensitive data.

Rationale:

Large volume commercial online retailers often store customer data in an RDBMS, but they use a data abstraction layer with limited privileges to access that data from their Web services and other externally-visible applications. This more fully protects the data in the database from unauthorized access and modification.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security

Evaluation Criteria:

1) Test:

Does the application protect sensitive data by using a data abstraction layer between the application and RDBMS?

Procedure:

Check that sensitive data is not readable and modifiable externally by the application.

Example:

BP1355

Do not design the database around the requirements of an application.

Rationale:

Databases often outlive applications (i.e., legacy databases and evolution of applications). Database can also support multiple applications. If design of the database were around the application, it may present security holes that other applications could exploit. It is better to design the application around the rules set by the database.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / RDBMS Security

Evaluation Criteria:

1) Test:

Is application business logic or rules not found in the database?

Procedure:

Make sure data validation is done at database even if it is already being done at the application level.

Example:

None

BP1360

Use the XML Infoset standard to serialize messages.

Rationale:

XML signatures rely on a character-by-character comparison for proper operations. A one character difference is a different result. So using a standard for serialization is very important to successful communications.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

Evaluation Criteria:

1) Test:

Does the Web service user serialize messages using the **XML** Infoset Standard?

Procedure:

Generate a test message and check it for compliance with the XML Infoset Standard.

Example:

None

2) Test:

Does the Web service provider serialize messages using the XML Infoset Standard?

Procedure:

Generate a test message and check it for compliance with the XML Infoset Standard.

Example:

None

BP1375

Use asymmetric encryption for sensitive SOAP-based Web services.

Rationale:

Most Web services exchange very few messages so the fact that asymmetric encryption is computationally intensive is a non-issue. Symmetric encryption is more efficient, but it is done by sharing a secret key outside the SOAP message communication which is less portable.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Design Tenet: Identity Management, Authentication, and Privileges
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Encryption and HAIPE
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / SOAP Security

BP1392

Register services in accordance with a documented service registration plan.

Rationale:

Program information services are provided via a shared space for use by consumers. In order to locate these services and access the corresponding information provided, the services should be registered in the **service registry** per direction of the shared information space manager.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Accessible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Design Tenet: Make Data Understandable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs
NESI / Part 2: Traceability / Naval Open Architecture / Reusability
NESI / Part 2: Traceability / Naval Open Architecture / Interoperability
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata Registry
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Policy / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata Registry
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata Registry
NESI / Part 5: Developer Guidance / Data / Metadata Registry

Evaluation Criteria:

1) Test:

Has the Program generated default service definitions and registered them in the DoD service registry?

Procedure:

Review that there is a service definition (URLs, WSDL entries, etc.) for each of the program information services and that they have been registered accordingly.

Example:

None

BP1394

Identify, publish and validate data objects exposed to the enterprise early in the data engineering process and update in a spiral fashion as development proceeds.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

BP1396

Develop high-level conceptual data models for new systems prior to Milestone A based on the business process context in which the system will be used.

Rationale:

An early high-level understanding of the data objects/entities involved in a system can help to clarify the purpose and context of the system and identify potential downstream interoperability issues.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

BP1397

Identify and develop use cases or reuse existing use cases as appropriate as early in the data engineering process as possible to support data model development.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data-Centric Publish-Subscribe (DCPS) / Reading/Writing Objects within a DDS Domain
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

BP1398

Develop Interaction models as appropriate.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

BP1400

Programs will use authoritative metadata established by the Joint Mission Threads (JMTs) when available.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

BP1408

Use a semantic description language such as Web Ontology Language (OWL) or Resource Definition Framework (RDF) to represent an Ontology.

Rationale:

Data producer recommendations are still maturing for how to handle data producers interaction with **Web Ontology Language** (**OWL**) or **Resource Definition Framework** (**RDF**).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata
NESI / Part 5: Developer Guidance / Data / Metadata

BP1409

Register Web services using Web Services Description Language (WSDL) and Universal Description, Discovery, and Integration (UDDI).

Rationale:

Ontology languages such as **Web Ontology Language** (**OWL**) or **Resource Definition Framework** (**RDF**) are currently immature.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata
NESI / Part 5: Developer Guidance / Data / Metadata

BP1567

Use the `<abbr>` and `<acronym>` tags to specify the expansion of acronyms and abbreviations.

Rationale:

Provides the user with easy access to the meaning of abbreviations and acronyms.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients

BP1568

Use a markup language to represent mathematical equations within Web pages.

Rationale:

Use a markup language such as MathML to display equations rather than creating images to display equations. This provides a more semantic meaning to those who may want to parse and use the equation and also provides for a more maintainable display of the equation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Browser-Based Clients
NESI / Part 5: Developer Guidance / User Interfaces / Browser-Based Clients

BP1594

Examine the use of Transmission Control Protocol (TCP) extensions and other transport protocols that have been designed to mitigate risk for high bandwidth, high latency satellite communications.

Rationale:

TCP performance over satellite links is generally poor due to delays and blockages inherent to satellite links. TCP extensions (e.g., **IETF** RFC 1323) and other transport protocols that have been developed to mitigate this risk should be considered for high bandwidth, high latency satellite communications.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Mobility
NESI / Part 4: Node Guidance / Node Transport / Mobility
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal

Evaluation Criteria:

1) Test:

If the system is involved in high bandwidth, high latency satellite communications, does the Node design address TCP performance?

Procedure:

Determine if parts of the system involve high bandwidth, high latency satellite communications and if so, look for a TCP extension.

Example:

None.

BP1597

Consider operational performance constraints in the design of the Node's Domain Name System (DNS).

Rationale:

Operational performance constraints such as narrow band width or intermittent service can have a large impact in how the **Domain Name System** (**DNS**) **server** is configured and consequently on the DNS chosen to support the Node.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)

Evaluation Criteria:

1) Test:

Have the operational performance constraints been delineated and used to justify the **Domain Name System** (**DNS**) used by the Node?

Procedure:

Review the acquisition documents looking for justifications for the selection of the Domain Name System (DNS).

Example:

None.

BP1614

Plan a contingency response to the Node becoming a new component service within another Node.

Rationale:

While the complexities of nested Nodes are currently not addressed within *NESI Part 4*, nested Nodes are a possibility; thus, Nodes should be prepared to interact in such an environment. Review, in order to do contingence planning, the guidance for Nodes in Part 4; analyze the operational tradespace and the impact on the Node architecture, on infrastructure interoperability, and on any relevant service standards. Prepare the Node for such interactions by encouraging the proper definition of key interfaces and capabilities and creating a distinction between Nodal infrastructure and component capabilities. These distinctions would allow a Node, for example, to supplant its own infrastructure with those of its new parent Node (either directly or via proxies).

Note: *The purpose of this practice is not necessarily to encourage nested Nodes, but to ensure that Nodes apply appropriate open* ***modular designs*** *both externally and internally to ensure greater interoperability in a variety of environments.*

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Client Platform
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange

Evaluation Criteria:

1) Test:

Does the Node use standardized interfaces to obtain the services of routine activities?

Procedure:

Look for alignment and adherence to guidance of NESI Part 4 and open systems approaches.

Example:

None.

BP1648

Host the Registration Web Service (RWS) registration portlet in the Node.

Rationale:

The process of registering a Node's **Component** service with the **Registration Web Service** (**RWS**) can be quite complicated. By providing access to the registration **portlet** the chances of obtaining a registration and of having valid data in the registration are greatly increased.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search

Evaluation Criteria:

1) Test:

Is the **Registration Web Service** (**RWS**) registration **portlet** hosted on the local Node?

Procedure:

Look for the Registration Web Service (RWS) registration portlet implementation.

Example:

None.

BP1649

Specifically include provisions for incremental implementation of the CES services.

Rationale:

The states of the individual services that comprise the **CES** are at different level of maturity. Consequently, an incremental approach allows Node development to continue in parallel with the CES functionality.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

Evaluation Criteria:

1) Test:

Is there an incremental development approach?

Procedure:

Review the Node's schedule for incremental development.

Example:

None.

BP1650

Specifically include provisions for incremental implementation of the hosting Node's CES services for Node Components.

Rationale:

The states of the individual services that comprise the **CES** are at different levels of maturity. Consequently, an incremental approach allows **Component** development to continue in parallel with the Node and CES functionality.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

Evaluation Criteria:

1) Test:

Is there an incremental development approach?

Procedure:

Review the schedule for Components for incremental development.

Example:

None.

BP1651

Ensure Node Components have access to Core Enterprise Services.

Rationale:

The burden of aligning to standard **CES** functionality and providing the functionality uniformly rests on the **Node** infrastructure, rather than the **components** within the Node. This isolates the components from the CES complexity and enhances portability and interoperability of the components. The access to CES may come from either from the standardized local Node infrastructure or through **Global Information Grid** (**GIG**) infrastructure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES and Intermittent Availability
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / CES and Intermittent Availability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity

Evaluation Criteria:

1) Test:

Do any **component** systems, applications or services implement any of the server side **CES Global Information Grid** (**GIG**) **Key Interface Profiles** (**KIPs**)?

Procedure:

Review the component systems, applications or services code for implementations of the server side CES Global Information Grid (GIG) Key Interface Profiles (KIPs).

Example:

None.

BP1653

Do not build dedicated Node guard products.

Rationale:

Current national policy dictates that a high-assurance guard or similar technology must be used whenever connecting networked security domains (i.e., ***SECRET US*** to ***SECRET REL*** or ***SIPRNET*** to ***NIPRNET***). Every single instantiation of every single guard needs to be approved by the appropriate authority. There are no type accreditations. Adding a new guard technique will likely incur additional scrutiny of the program as well as significant technical and schedule risks. The preferred approach is to use an already approved guard to mitigate risk.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Trusted Guards
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Trusted Guards

BP1654

Do not build dedicated Component guard products.

Rationale:

Current national policy dictates that a high-assurance guard or similar technology must be used whenever connecting networked security domains (i.e., ***SECRET US*** to ***SECRET REL*** or ***SIPRNET*** TO ***NIPRNET***). Every single instantiation of every single guard needs to be approved by the appropriate authority. There are no type accreditations. Adding a new guard technique will likely incur additional scrutiny of the program as well as significant and technical and schedule risks. The preferred approach is to use an already approved guard to mitigate risk.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Trusted Guards
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Trusted Guards

BP1661

Engage with the Net-Centric Enterprise Services (NCES) program office to explore approaches for mobile use of the Core Enterprise Services (CES) services in mobile Nodes that rely on Transmission Control Protocol/Internet Protocol (TCP/IP) for inter-node communication.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

BP1663

Design a Domain Name System (DNS) in coordination with the appropriate governing Internet Protocol Version 6 (IPv6) Transformation Office.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

BP1668

Acquire and configure approved guard products with the help of the Government program offices that acquire such guards.

Rationale:

Leveraging the certification documentation, expertise and existing relationships with the **National Security Agency** (NSA) and other pertinent authorities will streamline acquisition of approved guards.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Trusted Guards
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Trusted Guards

BP1669

Select XML-capable trusted guards.

Rationale:

As **XML** is a fundamental transfer format for data in interoperable net-centric environments, **trusted guards** should be capable of transferring XML data to facilitate cross-domain interoperability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Trusted Guards
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Trusted Guards
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange

BP1670

Plan for Black Core implementation in the local Node.

Rationale:

Node designers and operations personnel must implement and deploy encryptors or encryption support at enclave borders that can interoperate with partner Nodes and enclaves. See also the Black Core [P1152] and Confidentiality [P1340] perspectives.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Confidentiality / Black Core
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Confidentiality / Black Core
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Concurrent Transport of Information Flows

BP1671

Consider Black Core transition whenever there is a significant Node network design or configuration decision to make in an effort to avoid costly downstream changes caused by Black Core transition.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Confidentiality / Black Core
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Confidentiality / Black Core
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Concurrent Transport of Information Flows

BP1672

Be prepared to integrate fully with the Information Assurance (IA) infrastructure.

Rationale:

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Client Platform
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Net-Centric IA Posture and Continuity of Operations

BP1675

In the Node's Web infrastructure, support the technologies and standards used by the CES services under development as well as any technologies and standards used for Community of Interest (COI) services.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Infrastructure

BP1681

Make metrics for component services visible and accessible as part of the service registration and update the metrics periodically.

Rationale:

Metrics are normally also needed to ensure performance is provided according to more traditional **Service Level Agreements** (**SLAs**) and for operations management.

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Instrumentation for Metrics
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

BP1683

Coordinate the Node schedule with the schedules of the Core Enterprise Service (CES) providers.

Rationale:

An unavoidable consequence of the Node architecture is that Core Enterprise Services (CES) are evolving in parallel with the development of the Nodes themselves. If the schedule for a Node is not coordinated with those of the CES providers, newly deployed CES capabilities may not support Node capabilities under development.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

Evaluation Criteria:

1) Test:

Is there a Node roadmap that maps to the **Core Enterprise Services** (**CES**) schedules?

Procedure:

Look for a document that cross-references the Centric Enterprise Service schedules of capabilities to the Node's schedule.

Example:

None.

BP1684

Coordinate the Node schedule with the Component schedules.

Rationale:

All schedules are subject to slippage or modifications due to changing priorities. Changes in the development schedule for a Node"s capabilities can have an impact on the schedules of Node **components**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

BP1685

For Key Interface Profile (KIP) specifications that are not available or insufficiently mature, implement a "best effort" by following the published intent of functionality and monitor or participate in the relevant specification development body.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Key Interface Profile (KIP)
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Net-Ready Key Performance Parameter (NR-KPP) / Key Interface Profile (KIP)

BP1686

Align Node interfaces to Components for directory services with the guidance being provided by the Joint Directory Services Working Group (JDSWG) and sub-working groups, including such guidance as naming conventions, federation, and synchronization.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

BP1687

Follow Active Directory naming conventions defined in the *Active Directory User Object Attributes Specification* as required by the DoD CIO memorandum titled *Microsoft Active Directory (AD) Services*.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Directory Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Directory Services

BP1688

For Services Management, use an interim solution based on standardized Simple Network Management Protocol (SNMP) agents or other locally provided instrumentation and external monitoring tools.

Rationale:

An interim solution, until such time an enterprise instrumentation capability is available, will provide potential service consumers with real world historical performance metrics as well ensure support for negotiated **service level agreements** (**SLAs**). Example standards for performance instrumentation that enable enterprise-wide management include the Simple Network Management Protocol (SNMP, especially the Remote Network Monitoring or RMON specification), and Distributed Management Task Force (DMTF) standards.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Enterprise Management
NESI / Part 4: Node Guidance / Security and Management / Enterprise Management

BP1690

Use Node implemented Service Discovery (SD) for high availability.

Rationale:

One of the main reasons to develop a local Node **Service Discovery** (SD) Service is to support high availability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Service Enablers / Service Discovery
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 4: Node Guidance / Services / Service Enablers / Service Discovery

BP1691

Use Node implemented Service Discovery (SD) to meet compartmentalization needs.

Rationale:

For pilot implementations that are not reachable, such as might be the case in a higher classified environment, the Nodes should coordinate among themselves and DISA to provide pilot and full service implementations that are reachable.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / Service Enablers / Service Discovery
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Accessibility - Registered / Service Enablers / Service Discovery
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Service Discovery
NESI / Part 4: Node Guidance / Services / Service Enablers / Service Discovery

BP1692

Determine which Collaboration Service vendor offering to employ in a disadvantaged environment or separate network.

Rationale:

Monitor progress on fielding the NCES Collaboration Service. Performance or administration reasons may dictate hosting a collaboration solution at the Node.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Collaboration Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Collaboration Services

BP1693

Make sure that collaboration products used to satisfy urgent requirements are from the JTIC list.

Rationale:

See http://jitc.fhu.disa.mil/washops/jtcd/dcts/status.html and, for products certified for use on SIPRNET, http://jitc.fhu.disa.mil/washops/jtcd/dcts/projects.html), until the **Net-Centric Enterprise Services** (NCES) Collaboration Service is available.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Collaboration Services
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Collaboration Services

BP1695

Designate a Core Enterprise Services (CES) liaison to monitor the availability of services.

Rationale:

The **CES** liaison is an important role for keeping the Node and **component** engineering processes synchronized with CES providers such as **Net-Centric Enterprise Services** (**NCES**).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

BP1697

Make the parallel development of Core Enterprise Services CES outside the control of the Node a part of the Node's risk management activities.

Rationale:

Since the development of the **CES** is external to the development of the Node, there is an interdependency between the Node and the CES. The Node needs to consider this as an increase in the risk to the Node development. This risk needs to be communicated back to the CES management and development teams.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues

BP1698

Plan for the event that Component services within a Node cannot be invoked across security domains.

Rationale:

Until such approaches are prototyped and explored more fully, Nodes should anticipate that services will not be capable of cross-domain invocation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Overarching CES Issues / Cross-Domain Interoperation
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Cross-Security-Domains Exchange

BP1699

Configure routers in accordance with the Network Security Technical Implementation Guide (STIG).

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers

BP1700

Configure routers in accordance with Enclave Security Technical Implementation Guide (STIG).

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / IP Routing and Routers
NESI / Part 4: Node Guidance / Node Transport / Network Layer / IP Routing and Routers

BP1701

Configure Components for Information Assurance (IA) in accordance with the Network Security Technical Implementation Guide (STIG).

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Network Information Assurance
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Network Information Assurance
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Network Information Assurance
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Network Information Assurance
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Net-Centric IA Posture and Continuity of Operations

BP1702

Do not place services and information intended to be broadly accessible to other nodes behind a Virtual Private Network (VPN).

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Subnets and Overlay Networks / Virtual Private Networks (VPN)
NESI / Part 4: Node Guidance / Node Transport / Subnets and Overlay Networks / Virtual Private Networks (VPN)

BP1704

Consult the applicable Security Technical Implementation Guidance (STIG) documents as a fundamental part of design activities, and monitor the STIGs periodically for updates.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport
NESI / Part 4: Node Guidance / Node Transport

BP1705

Design Domain Name System (DNS) infrastructure in accordance with appropriate governing Internet Protocol Version 6 (IPv6) Transition Office requirements.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 4: Node Guidance / Node Transport / Network Services / Domain Name System (DNS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

BP1706

Design node networks, including the selection of Components and configuration, to support multicasting even if not currently used.

Rationale:

The use of multicasting is growing within the DoD and multicast capability is being actively engineered into the **Global Information Grid** (**GIG**).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Subnets and Overlay Networks / Broadcast, Multicast, and Anycast
NESI / Part 4: Node Guidance / Node Transport / Subnets and Overlay Networks / Broadcast, Multicast, and Anycast

BP1707

Configure and locate elements of the Node Web infrastructure in accordance with the Web Server Security Technical Implementation Guide (STIG).

Rationale:

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Infrastructure
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Computing Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Computing Infrastructure Integrity

BP1708

Configure and locate elements of the Node Web infrastructure in accordance with the Desktop Applications Security Technical Implementation Guide (STIG).

Rationale:

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Infrastructure
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Computing Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Computing Infrastructure Integrity

BP1709

Configure and locate elements of the Node Web infrastructure in accordance with the Network Security Technical Implementation Guide (STIG).

Rationale:

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Web Infrastructure
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Enterprise Security / Integrity / Computing Infrastructure Integrity
NESI / Part 4: Node Guidance / Security and Management / Enterprise Security / Integrity / Computing Infrastructure Integrity

BP1711

Use the CES Mediation Service, or a locally hosted copy, when XML document translation between schemas is a necessity.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Utility Services
NESI / Part 4: Node Guidance / Services / Utility Services

BP1712

Register developed mappings in the DoD Metadata Registry.

Rationale:

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Technical Architecture [now DISR]
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Utility Services
NESI / Part 4: Node Guidance / Services / Utility Services

BP1715

Design SCA log services according to the OMG Lightweight Log Service Specification.

Rationale:

One component of the SCA framework is a central logging facility, enabling the asynchronous collection of informational messages from any component connected to the framework; and the controlled read access to this information. The Lightweight Logging Service is a free-standing, self-contained service which is not connected to an event channel or similar infrastructure. Using a standard log service specification between SCA implementations can improve interoperability and portability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Communication Architecture
NESI / Part 5: Developer Guidance / Middleware / Software Communication Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: RF Acquisition

Evaluation Criteria:

1) Test:

Is the logging service designed according to the OMG Lightweight Log Service Specification? Is the logging service designed according to the OMG Lightweight Log Service Specification?

Procedure:

Check the log service provider's documentation for compliance with the OMG Lightweight Log Service Specification.

Example:

BP1716

Develop applications for SCA-compliant systems using a standard higher order language.

Rationale:

Developing SCA applications in higher order languages such as C enables independence from platform dependencies and helps ensure portability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Communication Architecture
NESI / Part 5: Developer Guidance / Middleware / Software Communication Architecture

Evaluation Criteria:

1) Test:

Does the application use a higher order language such as C rather than a lower order language such as Assembly?

Procedure:

Check what programming language is used to develop the SCA application.

Example:

BP1732

Follow the Upper Camel Case (UCC) naming convention for XML Type names.

Rationale:

The predominate style used by most programs or projects is to use the Upper Camel Case (UCC) for type names. Type names should be easy to differentiate from namespace prefixes and from attributes. Since the namespace prefix and the type name are separated by a non-whites character (i.e., the colon, :), it is easier to identify the type name from the namespace name if the type name follows the UCC.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Types
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas

Evaluation Criteria:

1) Test:

Do type names follow the Upper Camel Case (UCC) naming convention?

Procedure:

Examine the schema definition and verify that the type names follow the Upper Camel Case (UCC) name convention.

Example:

```
<xsd:complexType
   name="MyType"
   . . .
```

```
    </ xsd:complexType>
```

BP1733

Follow the Upper Camel Case (UCC) naming convention for XML element names.

Rationale:

The predominate style used by most programs or projects is to use the Upper Camel Case (UCC) for **XML element** names. Element names should be easily differentiable from namespace prefixes and from attributes. Since the namespace prefix and the element name are separated by a non-whites character (i.e., the colon, :), it is easier to identify the element name from the namespace name if the element name follows the UCC.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas

Evaluation Criteria:

1) Test:

Do element names follow the Upper Camel Case (UCC) naming convention?

Procedure:

Examine the schema definition and verify that the element names follow the Upper Camel Case (UCC) name convention.

Example:

BP1734

Follow the Lower Camel Case (LCC) naming convention for XML attributes.

Rationale:

The predominate style used by most programs or projects is to use the Lower Camel Case (LCC) for **XML attribute** names. Attributes are part of an attribute list which is a set of name="value" expressions separated by whitespace. Therefore, it is easy to find the beginning of the attribute name.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Defining XML Schemas
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Defining XML Schemas

Evaluation Criteria:

1) Test:

Do type names follow the Upper Camel Case (UCC) naming convention?

Procedure:

Examine the schema definition and verify that the type names follow the Upper Camel Case (UCC) name convention.

Example:

BP1739

Use the xsd qualifying prefix for XML Schema namespace.

Rationale:

Syntactically there is no reason why the XML Schema namespace can not be given any qualifier. However, for readability on the part of humans, using the xsd qualifier is clear, precise, concise and widely accepted.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces

Evaluation Criteria:

1) Test:

Does the XML schema use the xsd prefix for the XMLSchema namespace?

Procedure:

Look for the use of the XMLSchema namespace declaration and verify that the prefix is xsd.

Example:

The following is an example of using the xsd prefix for the XML Schema namespace:

```
<xsd:schema>
```

BP1741

Do not provide a schema location in import statements in schemas.

Rationale:

An import statement allows schema components from other schemas to be added to the current schema. The added schema components are associated with a namespace defined in the import statement. The import statement provides for the imported schema to also be optionally associated with a location where the schema can be found. Associating a schema location with a namespace during the import is referred to as early binding. This locks the definition to a specific implementation.

The following example highlights these points:

Weather Station Schema Definition

A weather station is defined as a collection of sensors with definitions that are to-be-determined.

Note: *The import of the http://www.Sensor.org without specifying the optional schema location.*

Note: *The use of the dangling type SensorType for the element Sensor. SensorType is bound later to a schema definition.*

```
<?xml version="1.0"?>
<xsd:schema
    xmlns: xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace=http://www.WeatherStation.org
    xmlns: s="http://www.Sensor.org"
    elementFormDefault="qualified">
 <xsd:import namespace="http://www.Sensor.org"/>
 <xsd:element name="WeatherStation">
   <xsd:complexType>
     <xsd:sequence>
      <xsd:element
           name="Sensor"
           type="s:SensorType"
           maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

Sensor Supplier Schema Definition

A sensor supplier creates a sensor specific definition for a sensor.

```
<?xml version="1.0"?>
<xsd:schema xmlns: xsd="http://www.w3.org/2001/XMLSchema"
                      targetNamespace="http://www.Sensor.org"
                      xmlns ="http://www.Sensor.org"
                      elementFormDefault="qualified">
   <xsd:simpleType name="SensorType">
     <xsd:restriction base="xsd:string">
         <xsd:enumeration value="barometer"/>
         <xsd:enumeration value="thermometer"/>
        <xsd:enumeration value="anenometer"/>
     </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

Weather Station Instance Document

A weather station instance document is created which binds the sensor suppliers definition of a sensor to the weather station. This allows the definition of the sensor to change or the location of the sensor definition (i.e. xsd) to change independently of the definition of the weather station.

```
<?xml version="1.0"?>
<ws:WeatherStation
   Xmlns: ws="http://www.WeatherStation.org"
   xmlns: xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation=
      "http://www.WeatherStation.org WeatherStation.xsd
       http://www.SensorSupplier.org SensorSupplier.xsd">
     <ws:sensor>thermometer</ws:sensor>
     <ws:sensor>barometer</ws:sensor>
     <ws:sensor>anenometer</ws:sensor>
</ws:WeatherStation>
```

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces

Evaluation Criteria:

1) Test:

Does the schema definition provide location for the imported schemas?

Procedure:

Examine the schema definition and make sure the schemaLocation attribute is not used in the import statement.

Example:

```
<xsd:import
  namespace="http://www.Sensor.org"
  schemaLocation=#Sensor.xsd#
/>
```

BP1742

Use the xsi qualifying prefix for XML Schema instance namespace uses.

Rationale:

Syntactically there is no reason why the XML Schema instance namespace can not be given any qualifier. However, for readability on the part of humans, using the xsi qualifier is clear, precise, concise and widely accepted.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Schema Documents / Using XML Namespaces
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Instance Documents
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Instance Documents

Evaluation Criteria:

1) Test:

Does the schema use the xsd prefix for the XMLSchema instance namespace?

Procedure:

Look for the use of the XMLSchema instance namespace declaration and verify that the prefix is xsi.

Example:

The following is an example of using the xsi prefix for the XML Schema instance namespace:

```
<xsd:schema xmlns: xsi="http://www.w3.org/2001/XMLSchema-instance">
```

BP1743

Use .xml as the file extension for files that contain XML Instance Documents.

Rationale:

By using the .xml extension for XML Instance Documents that are not associated with an application that requires another file extension (e.g., html, xslt):

- Readily identifies the file as containing XML to users
- Associates the XML file with various tools that work with XML Documents (i.e., browsers, parsers, validators, etc.)

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Visibility - Discoverable / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / XML Semantics / XML Instance Documents
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / XML Semantics / XML Instance Documents
NESI / Part 5: Developer Guidance / Data / XML / XML Semantics / XML Instance Documents

Evaluation Criteria:

1) Test:

Are there XML files that do not have the XML file extension or that are associated with specific applications?

Procedure:

Scan the files looking for files that contain XML that are not associated with an application. Examples of files that are associated with applications or services are .wsdl, .html, .htm and .xsl.

Example:

None.

BP1747

Use the xsl qualifying prefix for XSLT namespace.

Rationale:

Syntactically there is no reason why the XSLT namespace can not be given any qualifier. However, for readability on the part of humans, using the xsl qualifier is clear, precise, concise and widely accepted.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT

Evaluation Criteria:

1) Test:

Does the schema use the xsl prefix for the XSLT namespace?

Procedure:

Look for the use of the XSLT namespace declaration and verify that the prefix is xsl. Make sure there is only one namespace associated with the Transform XSD: http://www.w3.org/1999/XSL/Transform

Example:

The following is an example of using the xsl prefix for the XSL Transform namespace:

```
<xsl:stylesheet
xmlns: xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0"
    xmlns: xalan="http://xml.apache.org/xalan"
    xmlns: my-ext="ext1"
    extension-element-prefixes="my-ext">
```

BP1748

Separate static content from transformational logic in XSLTs.

Rationale:

Static XML content is content is copied verbatim from a static source, either internally or externally. Internal static content usually is found within the same input stream as the XSLT content. External static content is obtained from a different input stream and often comes from files or from data returned from a service.

Separating the static content from the transform logic facilitates maintenance by reducing the risk of unexpected side effects during the maintenance. In other words, maintenance to the transformational logic is isolated from the content. Content modifications have no affect on the transformation logic.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT

Evaluation Criteria:

1) Test:

Is static content imported using the xsl:copy element that selects a document?

Procedure:

Look for the intermixing of static content with the XSLT transform code.

Example:

BP1749

Use xsl:include for including XSL transforms.

Rationale:

Xsl:include includes other transforms and assigns the same precedence to the imported nodes as the importing document. This is the preferred method for including entire XSL transforms to allow for composition of multiple transforms into one that is much bigger.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT

Evaluation Criteria:

1) Test:

Procedure:

Example:

```
<xsl:include href="Guidance.xsl"/>
```

BP1750

Use xsl:import for reusing XSL code.

Rationale:

Since xsl:import includes other XSL code with a lower precedence than the importing document, it is best to just include small snippets of reusable XSL code. Also, xsl:import is inefficient versus xsl:include when dealing with large documents.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XSLT
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XSLT
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XSLT

Evaluation Criteria:

1) Test:

Procedure:

Example:

```
<xsl:import href="Guidance.xsl"/>
```

BP1752

Place dynamic XML element data within an XML CDATA section.

Rationale:

The content of dynamic data can not be predicted and could contain the XML special reserved characters < and & or the other characters that may cause parse errors; it is best to embed this data within an XML Character Data (CDATA) section that is ignored by parsers.

The following is an example of the use of a CDATA section that contains source code. Since the code could contain the < or & characters and be runtime dependent, a parse error could occur at runtime.
Please refer to the following example:

```
<![CDATA[
Public bool lessThan (a,b)
{ if (a!= null && b!=null a < b ) then
  { return true;
  } // End if
  else
  { return false;
  } // End else
} // End lessThan
]]>
```

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Syntax
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Syntax
NESI / Part 5: Developer Guidance / Data / XML / XML Syntax

Evaluation Criteria:

1) Test:

Do Element Data sections that are dynamically generated or are provided by external data surround the Element Data within a CDATA section?

Procedure:

Look for areas within XML instance documents or XML schemas that are candidates for dynamic content that can not be expected to be under the control of the XML instance document generator.

Example:

The following is an example of the use of a CDATA block that contains source code. Since the code could contain the < or & characters, a parse error could occur at runtime.
Please refer to the following example:

```
<![CDATA[
Public bool lessThan (a,b)
{ if (a < b ) then
  { return 1;
  } // End if
else
  { return 0;
  } // End else
} // End lessThan
]]>
```

BP1757

Do not ignore namespace prefixes in XPath expressions.

Rationale:

Ignoring namespaces can have undesired consequences. Some namespaces can contain nodes (elements) with the same name that contain different data structures. Consequently, if names bypass the use of the associated namespace, runtime errors can occur when attempts to process nodes of differing types occur.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XPath
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XPath
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XPath

Evaluation Criteria:

1) Test:

Do any XPath statements ignore namespaces?

Procedure:

Check for the existence of XPaths similar to the following:

```
//*[local-name()='location']
```

location is a node name defined in two different namespaces. For example, a geographic namespace may define location as latitude and longitude. It may also be defined in the display namespace as a x and y pixel coordinate.

Example:

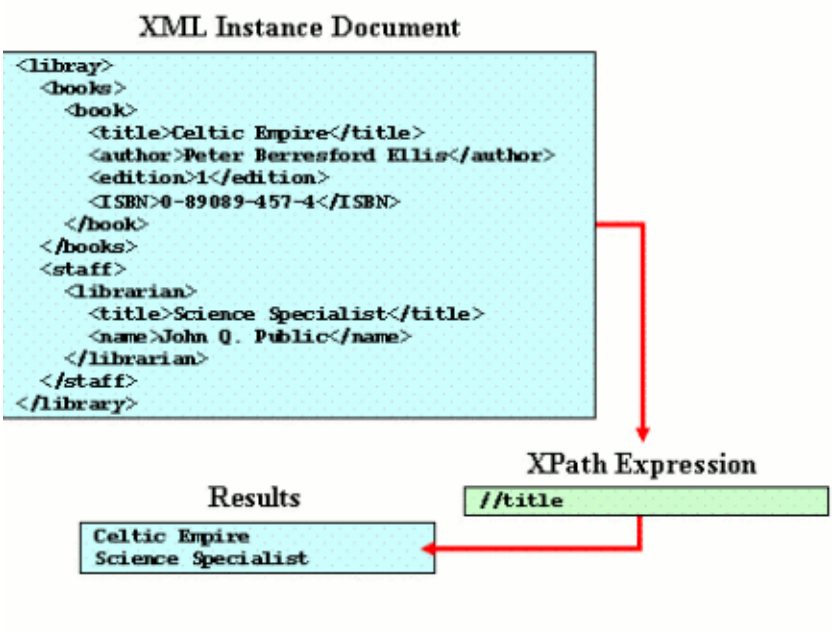
None.

BP1758

Make names in descendant expressions unique within an XML document.

Rationale:

The descendant operator, when misused, can have unintended consequences since nodes of the same name could possibly be included in multiple places in the XML Document. The XPath need to be written to eliminates unwanted nodes of the same name from other parts of the document.



I1172

In the above example, the <title> element can occur in multiple places within the document. Using the descendent operator '//' with the title element name returns all the titles.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / XPath
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / XPath
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / XPath

BP1764

Make all localizable user interface elements such as text and graphics externally configurable.

Rationale:

Externally configurable user interface elements allow for changing the supported language(s) at deploy-time or run-time without recompilation.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Internationalization

Evaluation Criteria:

1) Test:

Are all localizable presentation elements such as user interface text and graphics externally configurable?

Procedure:

Check for external configuration files for localizable presentation user interface elements.

Example:

BP1765

Declare the encoding type for all user interface content.

Rationale:

Declaring the encoding type allows for an application to determine the encoding type programmatically and make necessary display configuration settings at run-time. Also, for Unicode there are multiple ways to encode a character set.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Internationalization

Evaluation Criteria:

1) Test:

Do the user interface components (such as HTML pages) declare the encoding type?

Procedure:

Check to see that user interface components declare the encoding type.

Example:

Send the charset parameter in the Content-Type of HTTP header:

```
Content-Type: text/html; charset=utf-8
```

For XML (including XHTML), use the encoding pseudo-attribute in the XML declaration at the start of a document:

For HTML or XHTML served as HTML, use the tag inside :

BP1766

Develop user interfaces to accommodate variable syntactic structure for messages.

Rationale:

Different languages form sentence structures in different ways. Composing messages in code from multiple substrings in order to display the messages to the user may cause problems when porting the code to a language that uses a different sentence structure.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Designing User Interfaces for Internationalization
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Designing User Interfaces for Internationalization

Evaluation Criteria:

1) Test:

Are messages displayed on the user interface constructed in code using multiple substrings?

Procedure:

Check code for messages displayed to the user to see if the messages are composed from multiple substrings.

Example:

BP1767

Follow a standard process for human systems integration engineering such as the one defined by the International Organization for Standardization in ISO 13407:1999 on human-centered design processes for interactive systems.

Rationale:

Using a standard well-defined process increased the chance that required steps and procedures are completed during system development and lead to better usability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction

Evaluation Criteria:

1) Test:

Was a process for human systems integration followed during system development?

Procedure:

Look for documentation stating the human systems integration process.

Example:

BP1768

Use design patterns for application navigation.

Rationale:

Using common design patterns for application navigation builds on lessons learned, increases probability of user understand of the navigation pattern, and may result in better performance and a reduction in training.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User Interface Services / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 2: Traceability / DISR Service Areas / User (Physical/Cognitive) / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces
NESI / Part 5: Developer Guidance / User Interfaces / Human-Computer Interaction / Human Factor Considerations for Web-Based User Interfaces

Evaluation Criteria:

1) Test:

Does the application navigation follow design patterns?

Procedure:

Identify the design patterns used for application navigation.

Example:

- Use a hub navigation pattern for tasks that consist of multiple independent steps performed in any order
- Use wizard navigation pattern for tasks that consist of multiple interdependent steps that are defined in a predefined order.
- Use a pyramid navigation pattern when it is necessary to navigate to sibling, child, or parent pages while completing tasks.

BP1769

Provide wrapper or adapter classes to isolate XML parser implementations.

Rationale:

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / XML / XML Processing / Parsing XML
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / XML / XML Processing / Parsing XML
NESI / Part 5: Developer Guidance / Data / XML / XML Processing / Parsing XML

BP1790

Stipulate that the Offeror is to describe how the proposed technical solution reuses services or demonstrates composeability and extensibility by building from existing reusable components and/or services.

Rationale:

Reuse of existing components and services leads to reduced costs and promotes modularity and composeability. Reusable artifacts are common in large distributed networks. Future systems will be required to demonstrate composing new solutions from reusable components and services.

Referenced By:

NESI / Part 6: Contracting Guidance for Acquisition / Contracting Guidance for Reuse / Section L: Instructions, Conditions, and Notices to Offerors
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Layering and Modularity

Evaluation Criteria:

1) Test:

Does the Offeror demonstrate reuse of existing components or services?

Procedure:

Identify in the proposal the components or services identified as being reused.

Example:

None.

BP1811

Isolate all use of vendor specific extensions to the Data Distribution Service (DDS).

Rationale:

Vendor specific extensions may be required to perform certain configuration actions, take advantage of features that are in the process of becoming standard (e.g., version 1.3, expected to be adopted by late 2007), or simply use additional capabilities provided by a vendor that would otherwise require significant application work.

Vendor-specific extensions should only be used if there is no standard API from the DDS specification that accomplishes the same function.
One method of isolating vendor-specific extensions is to enclose the code within conditional compile instructions (e.g., **`#ifdef #endif`** for C/C++) such that portability is not compromised.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

Evaluation Criteria:

1) Test:

Does the implementation use wrappers or façade patterns to isolate vendor specific code?

Procedure:

Is vendor specific code contained within a limited number of classes or objects?

Example:

None

2) Test:

Does the implementation annotate vendor specific code?

Procedure:

Look for the use of compiler instructions that isolate vendor specific code.

Example:

```
#ifdef DDS_VENDOR_XXXX
#. <vendor specific code
#endif
```

BP1812

Use the `RELIABILITY` Quality of Service (QoS) kind `BEST_EFFORT` for Data Distribution Service (DDS) Topics that are written frequently where missing an update is not important because new updates occur soon thereafter.

Rationale:

The use of the **`RELIABILITY`** QoS kind **`BEST_EFFORT`** allows the middleware to use a lower-latency, lighter-weight protocol to send data that avoids the need for extraneous Acknowledgement and Heartbeat traffic. This protocol also exploit multicast more efficiently because there is never a need to send any acknowledgments back to the sender. Consequently, this protocol should be preferentially used whenever the nature of the Topic is such that occasionally missing a message has no adverse consequence to the system.

Data that is continually published and represents updates to data-objects or where only the most current value is of interest to the system are prime candidates for **`BEST_EFFORT`** communication.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`RELIABILITY`** QoS selection properly justified for each Topic? Is **`BEST_EFFORT`** kind used whenever the nature of the Topic allows it?

Procedure:

Review the system documentation for proper justification of the **`RELIABILITY`** QoS assigned to each Topic.

Example:

None

BP1813

Use the `RELIABILITY` Quality of Service (QoS) kind `RELIABLE` for Data Distribution Service (DDS) Topics written sporadically or where it is important that the current data in the Topic is received reliably.

Rationale:

The **`RELIABILITY`** QoS kind **`RELIABLE`** ensures the service will make all necessary attempts to deliver the information. The DDS protocol employs Heartbeats and Acknowledgments to accomplish this task.

Data that is rarely written or which the system requires never to be lost should be published with **`RELIABILITY`** QoS kind **`RELIABLE.`**

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`RELIABILITY`** QoS selection properly justified for each Topic? Is **`RELIABLE`** kind used whenever the nature of the Topic requires it?

Procedure:

Review the system documentation for proper justification of the **`RELIABILITY`** QoS assigned to each Topic.

Example:

None

BP1814

Use the `DEADLINE` Quality of Service (QoS) to for Data Distribution Service (DDS) DataWriters for which data is published at a constant rate.

Rationale:

The frequency with which a particular data-object is updated may affect the logic of the overall system. For example some radar processing algorithms may have been written under the assumption that each track is updated every five seconds after the radar completes a new sweep.

If the **`DataWriter`** specifies a **`DEADLINE`** QoS, DDS can monitor that each data-object is indeed written at least once per stated period. Furthermore, DDS can propagate the **`DataWriter`** deadline to the **DataReaders** such that they can realize whether their expectation matches what the **`DataWriter`** provides. If the expectation cannot be met the application is notified of an incompatible QoS.

By using this QoS the modules can remain de-coupled, yet provide the essential information required for the integrated system to operate as expected.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`DEADLINE`** QoS used in all the **`DataWriters`** where it could?

Procedure:

Review the system documentation for proper justification of the **`DEADLINE`** QoS assigned to each **`DataWriter`**.

Example:

BP1815

Use the `DEADLINE` Quality of Service (QoS) for Data Distribution Service (DDS) DataReaders that expect data to be sent to them at a constant rate.

Rationale:

The frequency with which a particular data-object is updated may affect the logic of the overall system. For example some radar processing algorithms may have been written under the assumption that each track is updated every five seconds after the radar completes a new sweep.

If the **`DataReader`** specifies a **`DEADLINE`** QoS then DDS can monitor that an update to each data-object is indeed received at least once per stated period and if not notify the application. Furthermore, DDS can propagate the **`DataReader`** deadline to the **DataWriters** such that they can realize whether they can meet the expectation of the **`DataReader`**. If the expectation cannot be met the application is notified of an incompatible QoS.

By using this QoS the modules can remain decoupled, yet provide the essential information required for the integrated system to operate as expected.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`DEADLINE`** QoS used in all the **`DataReaders`** where it could?

Procedure:

Review the system documentation for proper justification of the **`DEADLINE`** QoS assigned to each **`DataReader`**.

Example:

BP1816

Use the `LIVELINESS` Quality of Service (QoS) for Data Distribution Service (DDS) Topics where data is not sent sporadically; that is, it is sent with no fixed period.

Rationale:

Some data (e.g., alarms or commands) are sent without a fixed period. In these cases the fact that updates are not received could indicate that there is either no new data, or alternatively that there is a system malfunction and the writer is not able to send the data. The DDS **`LIVELINESS`** QoS allows the application to discern between these two situations.

Setting the **`LIVELINES`** QoS indicates to DDS that in the event that there is no data to send, periodic liveliness messages should be exchanged to notify the **DataReader** that the **DataWriter** is still active, capable of communication, and therefore that if it receives no data then it is in fact because there is none to send. The DDS monitors the **`LIVELINESS`** and informs the application when a **`DataWriter`** loses its ***liveliness*** via the proper status message dispatched to the Listener.

Proper settings of the **`LIVELINESS`** QoS is also required to receive proper **`InstanceState`** information with the received Samples as well as to manage **`OWNERSHIP`** in the presence of failures.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Are all **`DataWriters`** or **`DataReaders`** that do not set a **`DEADLINE`** setting a **`LIVELINESS`**?

Procedure:

Check the QoS used to create **`DataReaders`** and **`DataWriters`** and ensure that if the **`DEADLINE`** QoS is not set, then the **`LIVELINESS`** QoS is set to a non-infinite value

Example:

BP1817

Use the `MANUAL_BY_TOPIC` setting of the `LIVELINESS` Quality of Service (QoS) for Data Distribution Service (DDS) Topics where the presence and health of the DataWriter is critical to the proper operation of the system.

Rationale:

Certain Topics are monitoring functions so critical to the health of the system that reliance on the health of the process that writes the Topic does not offer sufficient assurance that the application is performing the proper monitoring functions. In these situations the **`MANUAL_BY_TOPIC`** setting of the **`LIVELINESS`** QoS requires the **`DataWriter`** to either write the data at least once per liveliness period or invoke the **`DataWriterasset_liveliness()`** operation to indicate proper functioning.

The **`MANUAL_BY_TOPIC`** setting of the **`LIVELINESS`** QoS can be thought of as the distributed system equivalent to the mechanical dead man's switches used to monitor that the operator of a system (e.g., a train locomotive) is still present and able to function.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Are all critical **`DataWriters`** either setting a deadline or using a **`LIVELINESS`** set to **`MANUAL_BY_TOPIC`**?

Procedure:

Check the QoS used to create **DataReaders** and **`DataWriters`** and ensure that if the **`DEADLINE`** QoS is not set, then the **`LIVELINESS`** QoS is set to **`MANUAL_BY_TOPIC`** and has a non-infinite value.

Example:

BP1818

Use the `HISTORY` Quality of Service (QoS) kind `KEEP_LAST` for Data Distribution Service (DDS) Topics that represent system state, in that new data-values replace the old values for each Keyed data-object.

Rationale:

Some Topics represent system state. The readers of the Topic need only know the most current value (or last set of N values) of each data-object published under the Topic. An example of this may be a Topic representing the reading of different temperature sensors. Applications only care to read the most recent value of each sensor. The same may be said of a Topic representing the expected arrival times of aircraft at a given airport.

The **`HISTORY`** QoS setting of **`KEEP_LAST`** indicates to the middleware that it should not attempt to store or propagate old values of data objects; instead, only the most recent value(s) are of interest. This allows DDS to conserve system resources (memory) as well as to save the bandwidth required to send information that is no longer relevant. Reader applications also benefit as they do not waste time reacting to data values that are no longer current.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`HISTORY`** QoS properly sent on all Topics?

Procedure:

Check the QoS used to create **DataReaders** and **DataWriters** and check how the **`HISTORY`** QoS is set. Ensure that a kind **`KEEP_LAST`** is used whenever the Topic represents system state.

Example:

BP1819

Use the `HISTORY` Quality of Service (QoS) kind `KEEP_ALL` for Data Distribution Service (DDS) Topics that represent events or commands where all values written should be delivered to the readers (i.e., new values do not replace old values).

Rationale:

Some Topics represent events, commands, or messages in that new data written never replaces previously-written values, rather they should all be delivered to the **DataReader**.

The **`HISTORY`** QoS setting of **`KEEP_ALL`** indicates to the middleware that it should not replace old values with new values on the topic. Subject to other QoS (such as filters, ownership, lifespan) they should all be delivered to the **`DataReaders`**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`HISTORY`** QoS properly sent on all Topics?

Procedure:

Check the QoS used to create **`DataReaders`** and **DataWriters** and check how the **`HISTORY`** QoS is set. Ensure that a kind **`KEEP_ALL`** is used whenever the Topic represents 'events', commands or messages.

Example:

BP1820

Use `TIME_BASED_FILTER` Quality of Service (QoS) to protect DataReaders that cannot handle all the traffic that could be written by the writers on that Data Distribution Service (DDS)Topic and just need periodic updates on the most current data-values.

Rationale:

The **`TIME_BASED_FILTER`** QoS allows a **`DataReader`** to specify that it is interested only in (potentially) a subset of the values of the data. The filter states that the **`DataReader`** does not want to receive more than one value each minimum_separation, regardless of how fast the changes occur. The default setting is minimum_separation=0 indicating that the **`DataReader`** is potentially interested in all values.

In heterogeneous systems, it is common that some subsystems either cannot handle or do not choose to handle all the information available on a Topic. For example a high-level display at an airport control tower may not need to update the location of aircraft more often than each second as the human operators looking at the display would not be able to take advantage of faster refreshes. Nevertheless, the data is published at much higher rate to allow for algorithmic processing on other subsystems.

By setting the **`TIME_BASED_FILTER`** properly an application that has a well defined maximum refresh rate can protect itself from system reconfigurations which may result in a Topic being published faster than originally anticipated.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`TIME_BASED_FILTER`** QoS properly sent on all **`DataReaders`**?

Procedure:

Check the QoS used to create **`DataReaders`** and check whether the **`TIME_BASED_FILTER`** QoS is set. Ensure it is set to a proper non-zero minimum_separation whenever the application can be in a system where it is not expected to handle all the updates on the Topic.

Example:

BP1821

Use the Data Distribution Service (DDS) `LIFESPAN` Quality of Service (QoS) to indicate that data is only valid for a finite time period and stale data is discarded after a certain expiration time elapses.

Rationale:

Some **Topics** represent data with a natural expiration. For example the location of an aircraft during flight becomes less relevant as the information ages and may not have any tactical value after a certain time elapses.

The setting of the **`LIFESPAN`** QoS indicates to DDS the maximum time duration during which the information is relevant. After this time elapses, DDS is no longer required to maintain the information or provide it to the **DataReaders**. Proper setting of this QoS can therefore save resources and bandwidth as well as save **`DataReaders`** from being notified of information that is no longer relevant.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`LIFESPAN`** QoS properly sent on all Topics?

Procedure:

Check the QoS used to create **DataWriters** and check whether the **`LIFESPAN`** QoS is set. Ensure it is set to a proper non-infinite duration whenever appropriate.

Example:

BP1822

Use the `PARTITION` Quality of Service (QoS) to limit the scope of the data written/read on a Data Distribution Service (DDS) Topic to only the writer/readers that have a common partition.

Rationale:

The **`PARTITION`** QoS is used to introduce logical partitions within a Topic. A **DataWriter** only communicates with a **DataReader** if (in addition to matching the Topic and having compatible QoS) they share a common partition

The **`PARTITION`** QoS is set on the **Publisher** and **Subscriber** and affects all the **`DataWriters`** in the Publisher and **`DataReaders`** on the Subscriber.

The **`PARTITION`** QoS can be used to introduce a logical scope and the fact that it is adjustable at run-time makes it possible to perform system reconfigurations. For example, a **`DataReader`** could be temporarily isolated from the rest of the system by switching its Partition to something that nobody matches. Similarly a **`DataWriter`** and **`DataReader`** could be reconfigured to have an "isolated session" by switching to a partition that nobody else uses.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`PARTITION`** QoS used to simplify application logic where appropriates?

Procedure:

Check the QoS used to create Publisher and Subscriber and check whether the **`PARTITION`** QoS is used. Verify that the application does not use some other non-standard way to implement a use-case that could be supported using the **`PARTITION`** QoS.

Example:

BP1823

Use the Data Distribution Service (DDS) `RESOURCES_LIMITS` Quality of Service (QoS) in platforms with limited memory or in real-time systems to properly configure the resources that will be utilized and avoid exhaustion of system resources at run-time.

Rationale:

The **`RESOURCE_LIMITS`** QoS on the **DataWriter** and **DataReader** specifies the resources that DDS can consume in order to meet the requested QoS.

While these limits can be left to their default "auto-grow" settings proper configuration of these limits is important in any system that has limited resources and is expected to operate reliably for long time spans. By setting the limits the developer can balance the resources consumed for each topic and protect the system against a mis-configuration when a **Topic** that produces too much data exhausts the resources needed to manage other Topics. This is especially important if other QoS do not limit the amount of data that the system would need to store (e.g. if **`HISTORY`** is set to **`KEEP_ALL`** and **`LIFESPAN`** is set to infinite).

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`RESOURCE_LIMITS`** QoS set on the **`DataWriter`** and **`DataReader`**?

Procedure:

Check the QoS used to create **`DataWriters`** and **`DataReaders`** and check whether the **`RESOURCE_LIMITS`** are set to some finite limits. Ensure that any **`DataWriters`** and **`DataReaders`** that have if **`HISTORY`** kind **`KEEP_ALL`** and **`LIFESPAN`** duration set to infinite use the **`RESOURCE_LIMITS`** to control the maximum resource utilization.

Example:

BP1824

Use the `USER_DATA` Quality of Service (QoS) to communicate metadata on the DomainParticipant that may be used to authenticate the application trying to join the Data Distribution Service (DDS) Domain.

Rationale:

In many cases the application needs to send additional information that describes the **`DomainParticipant`** to other participants in the DDS Domain. This information can be used to authenticate the participant or to meet any other application-specific need.

The **`USER_DATA`** QoS on the **`DomainParticipant`** allows the application to store un-interpreted bytes that will be propagated via the DDS built-in discovery mechanism and will be accessible to the other **`DomainParticipants`** on the system.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`USER_DATA`** QoS set on the **`DomainParticipant`**?

Procedure:

Check the creation of the **`DomainParticipant`** and determine whether the **`USER_DATA`** QoS is used. Ensure that the application does not use another non-standard way to accomplish the same function.

Example:

None.

BP1825

Use the `ignore_participant` operation on the DomainParticipant to deny access to another DomainParticipant trying to join a Data Distribution Service (DDS) Domain.

Rationale:

The **`ignore_participant`** operation can be used by a **`DomainParticipant`** to prevent another **`DomainParticipant`** from communicating with the first participant. In combination with the **`USER_DATA`** QoS on the participant this mechanism can be used to authenticate **`DomainParticipants`**.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

Evaluation Criteria:

1) Test:

Is the **`ignore_participant`** operation used whenever there is a requirement to prevent arbitrary participants from accessing the information the first participant publishes or subscribes?

Procedure:

Check the code for any occurrences of the **`ignore_participant`** operation.
Ensure that the application does not use another non-standard way to accomplish the same function.

Example:

BP1826

Use the `USER_DATA` Quality of Service (QoS) on the DataWriters and DataReaders to communicate metadata that may provide application-specific information of the entity writing/reading data in a Data Distribution Service (DDS) Domain.

Rationale:

In many cases the application needs to send additional information that describes the **`DataWriter`** or the DataReader to other entities in the DDS Domain. This information can be used to authenticate the **`DataWriter`**/ Reader or to meet any other application-specific need.

The **`USER`**_DATA QoS on the **`DataWriter`** and the **`DataReader`** allows the application to store un-interpreted bytes that will be propagated via DDS's built-in discovery mechanism and will be accessible to the other **`DataWriters`** and **`DataReaders`** on the system.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

Evaluation Criteria:

1) Test:

Is the **`USER_DATA`** QoS set on the **`DataWriter`** and **`DataReader`**?

Procedure:

Check the creation of the **`DataWriter`** and **`DataReader`** and determine whether the **`USER_DATA`** QoS is used. Ensure that the application does not use another non-standard way to accomplish the same function.

Example:

None.

BP1827

Use the `ignore_publication` and `ignore_subscription` on the DomainParticipant to deny access to a Data Distribution Service (DDS) Topic by a specific DataWriter or DataReader.

Rationale:

The ignore_publication and ignore_subscription operation can be used by a **`DomainParticipant`** to prevent a **`DataWriter`** or **`DataReader`** from communicating with the entities in the participant. In combination with the **`USER_DATA`** QoS on the **`DataWriter`** and **`DataReader`** this mechanism can be used to check that the **`DataWriter`** and **`DataReader`** have the proper **access control** to the Topic.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

Evaluation Criteria:

1) Test:

Are the **`ignore_publication`** and **`ignore_subscription`** operation used whenever there is a requirement to prevent arbitrary **`DataWriters`** or **`DataReaders`** from accessing the information on a Topic?

Procedure:

Check the code for any occurrences of the **`ignore_publication`** and **`ignore_subscription`** operation. Ensure that the application does not use another non-standard way to accomplish the same function.

Example:

BP1828

Use the Data Distribution Service (DDS) `OWNERSHIP` Quality of Service (QoS) kind set to `SHARED` when each unique data-object within a DDS Topic to which multiple DataWriters can write.

Rationale:

A primary intent of DDS is to support a loosely coupled publish and subscribe paradigm where the publishing is isolated from subscribing through autonomous topics. As a result, an implementation that requires a single data publisher currently may evolve to require multiple data publishers in the future. By using a **`OWNERSHIP`** QoS kind set to **`SHARED`** and allowing the DDS infrastructure to connect the **publisher** and the **subscriber** together, the implementation may be extended to another DDS profile without having to modify the original source code.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service

BP1829

Use the Data Distribution Service (DDS) `OWNERSHIP` Quality of Service (QoS) kind set to `EXCLUSIVE` when multiple DataWriters cannot write each unique data-object within a DDS Topic simultaneously.

Rationale:

DDS easily supports multiple **publishers** adding data to the same topic without impacting the **subscribers**. Using the DDS **`OWNERSHIP`** QoS kind set to **`EXCLUSIVE`** places the entire burden off supporting the multiple publishers on the DDS implementation rather than the publisher or subscriber code. This results in an increase of modularity, portability and the maintainability.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Quality of Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Layering and Modularity

BP1830

Use the Data Distribution Service (DDS) Content Profile to tailor subscription message data.

Rationale:

The DDS Content Profile allows for the **subscribers** to select and refine the data that is retrieved from a **Topic**. This tailoring code is part of the DDS infrastructure and is well tested and reliable. Not using the DDS Content Profile and using code within the subscriber increases the complexity of the subscriber and causes tight coupling between the subscriber code and the Topic.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity

BP1831

Use the Data Distribution Service (DDS) Persistence Profile to ensure durable data delivery.

Rationale:

The DDS Persistence Profile allows for data persistence within a **Topic** independent of hardware platform and operating system (OS) and to retrieve the data using the standard **Structured Query Language** (SQL). As a result, the publisher, subscriber and the topic remain loosely coupled from each other as well as the hardware platform or the OS.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / Decoupling Using DDS and Publish-Subscribe

BP1832

Handle all Data Distribution Service (DDS) Data Local Reconstruction Layer (DLRL) Exceptions.

Rationale:

The DLRL API may raise Exceptions under certain conditions. The following is an extensive list of all possible Exceptions and the conditions in which they will be raised:

`DCPSError`	If an unexpected error occurred in the DCPS
`BadHomeDefinition`	If a registered **`ObjectHome`** has dependencies to other, unregistered **`ObjectHomes`**.
`NotFound`	If a reference is encountered to an object that has not (yet) been received by the **DCPS**.
`AlreadyExisting`	If a new object is created using an identify that is already in use by another object.
`AlreadyDeleted`	If an operation is invoked on an object that has already been deleted
`PreconditionNotMet`	If a precondition for this operation has not (yet) been met.
`NoSuchElement`	If an attempt is made to retrieve a non-existing element from a Collection.
`SQLError`	If an **SQL** expression has bad syntax, addresses non-existing fields or is not consistent with its parameters.

Note: *DLRL, a recent addition to the DDS specification is particularly rich; implementations using this upper level profile of the specification are still emerging.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)

BP1833

Use the Data Distribution Service (DDS) Object Model Profile for accessing message data as objects.

Rationale:

The DDS **Data Local Reconstruction Layer** (DLRL) is intended to provide an abstraction layer between the actual underlying data and the higher level object level concepts used in applications. The Object Model Profile defines how applications interact with the abstract object layer. Applications that are bound directly to the actual underlying data are tightly coupled to the layer and are subject to its evolutionary changes.

Note: *DLRL, a recent addition to the DDS specification is particularly rich; implementations using this upper level profile of the specification are still emerging.*

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Messaging / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Messaging / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)
NESI / Part 5: Developer Guidance / Middleware / Messaging / Data Distribution Service (DDS) / DDS Data Local Reconstruction Layer (DLRL)

BP1837

Update the net-centric and SOA migration plan in an iterative manner as the program gains migration experience and conditions change.

Rationale:

Most large-scale net-centric and SOA migrations are expected to be lengthy and subject to many influencing and changing factors. As a result, they should be implemented in phases. Small-scale migrations may be able to execute the bulk of the migration in a single increment, but the migration plan should still be revisited for potential updates over time. Specifically, use the same methodology for creating updates to the plan as for creating the initial baseline version.

Referenced By:

NESI / Part 3: Migration Guidance / Migration Planning Process
NESI / Part 3: Migration Guidance / Migration Planning Process / Plan Migration / Finalize Migration Plan
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

Evaluation Criteria:

1) Test:

Does the migration plan track its currency date and any updates?

Procedure:

Examine the migration plan for a currency date and update tracking.

Example:

None.

BP1840

Identify opportunities to apply the principles of net-centricity and SOA throughout the course of the program.

Rationale:

All of the program's modernization activities have the potential to include opportunities to migrate to net-centricity and SOA. Even requirements that on the surface appear to not relate to net-centricity or SOA may contain a net-centric or SOA aspect. Coordinate with both user and developer personnel to identify these opportunities and the associated risks. Be careful to not overstate the requirements.

Referenced By:

NESI / Part 3: Migration Guidance / Migration Planning Process / Assess Migration Needs / Assess As-Is Requirements
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

Evaluation Criteria:

1) Test:

Does the program's migration plan describe an approach for identifying opportunities to apply net-centric and SOA principles throughout the course of the program?

Procedure:

Verify that the migration planning documentation contains a description of an approach for identifying net-centric and SOA migration opportunities.

Example:

None.

2) Test:

Does the program's migration plan contain an analysis of opportunities to apply net-centric and SOA principles throughout the course of the program?

Procedure:

Review the program's migration planning documentation and verify that it contains an analysis of opportunities of opportunities to apply net-centric and SOA principles throughout the course of the program.

Example:

None.

BP1845

Consider key enterprise-level concerns when planning and executing a migration to net-centricity and SOA.

Rationale:

The complexity of migration planning and execution requires careful consideration of numerous factors. Early and deliberate consideration of these factors is required to successfully achieve both program and enterprise-level objectives associated with the migration.

Referenced By:

NESI / Part 3: Migration Guidance / Critical Migration Concerns
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Network Connectivity
NESI / Part 3: Migration Guidance / Migration Planning Process / Plan Migration / Develop Implementation Plans

Evaluation Criteria:

1) Test:

Does the implementation plan for net-centricity and SOA migration contain considerations for key enterprise-level concerns?

Procedure:

Review the migration plan tasks and verify that they address critical migration concerns.

Example:

None.

BP1855

Identify types of data items for potential sharing external to the program.

Rationale:

Identifying the types of data items that may be shared external to the program will drive the refinement of interoperability requirements and the design of interoperability mechanisms. Potential sources for this information include descriptions of existing data stores and existing or planned interfaces, architectural products, data models, document repositories, etc. Consider the logical entities represented by the data. Consider issues related to security classification, frequency of exchange, and file formats. Consider issues related to timeliness and data quality.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1856

Identify specific data items for potential sharing external to the program.

Rationale:

Identifying the specific data items that may be shared external to the program will drive the refinement of interoperability requirements and the design of interoperability mechanisms. Potential sources for this information include descriptions of existing data stores and existing or planned interfaces, architectural products, data models, document repositories, etc. Identify the source, typical destinations, security classification, frequency of exchange, and typical size of the data. Avoid sharing data from other sources as a "pass through.."

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1857

Prioritize data items for potential sharing external to the program.

Rationale:

Prioritizing data items for potential sharing external to the program will support the planning of the migration to include the allocation of development resources. Analyze key operational processes to identify operationally important information exchanges. Consult with **Communities of Interest** (COIs) to determine the demand for specific data assets. Consider such factors as cost, time, and engineering difficulty.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1858

Publish preliminary program data-related development plans.

Rationale:

While initially incomplete, preliminary program data-related development plans may prove useful to other programs as they plan their migrations due to the inherent interdependencies introduced by the Net-Centric Data Strategy. Create initial descriptions of data items that are forecast to be sharable using the **DoD Discovery Metadata Specification** (DDMS) and publish them in the **DoD Metadata Registry**.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1859

Create external representations for sharable data items.

Rationale:

External representations will drive the implementation of both providers and consumers of the data items. Coordinate both internally within the program and externally with appropriate **COI**s. Explore de facto loose coupler and existing COI data formats. Create **XML schema** definitions for the data items and publish them in the **DoD Metadata Registry**.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1860

Create metadata representations for sharable data items.

Rationale:

Metadata representations will drive the implementation of both providers and consumers of the data items. Identify what data items will be searchable taking into account cost and performance considerations. Tag individual data items as appropriate using automated metadata generation where possible. Use the **DoD Discovery Metadata Specification** (DDMS).

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1861

Publish data access services that implement interfaces to shared data.

Rationale:

Services make data accessible using standardized mechanisms and enable the loose coupling of systems that process data. Select the appropriate underlying SOA-based technologies using NESI. Design service interfaces using the **XML schema** definition for the data exchange. Take into account security, performance, and versioning considerations. Use the **DoD Discovery Metadata Specification** (DDMS) and the **DoD Metadata Registry**. Test, deploy, and sustain data exchange mechanisms that support the NCDS in much the same fashion as any other mission-oriented software. The standard lifecycle methodologies used for other systems and software will apply.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)

BP1863

Make shareable data assets visible, even if they are not accessible.

Rationale:

Making data visible using a consistent, standardized metadata specification within a Net-Centric Environment (NCE) facilitates a federated cross-organizational discovery capability [R1172]. A common specification for the description of information allows for a comprehensive capability that can locate all information across the NCE regardless of format, type, location, or classification, dependent on user authorization. The **DoD Metadata Specification** (**DDMS**) was developed to support Enterprise-wide data discovery by providing a common set of descriptive metadata elements. Discovery metadata must conform to the DDMS in accordance with DoD Directive (DoDD) 8320.2 [R1217]. Information owners tag information with DDMS-compliant metadata to ensure discoverability of information in the NCE.
The extensible nature of the DDMS supports domain-specific or **COI** discovery metadata requirements and extends the element categories identified in the DDMS Core Layer used to describe information. Use of the DDMS does not preclude use of other metadata processes or standards. For example, record-level database tagging and in-line document tagging are common practices to support various department objectives. These tagging initiatives should be enhanced to include the DDMS for enterprise discovery.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Service-Oriented Architecture (SOA)
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: IPv6

Evaluation Criteria:

1) Test:

Does the system provide discovery metadata in accordance with the DoD Discovery Metadata Standard (DDMS) for all data posted to shared spaces?

Procedure:

Examine the DoD Metadata Registry for program/system.

Example:

Discoverable information has associated DDMS metadata that can be found in the DDMS).

BP1864

Layer architectures to support clear boundaries between data management, presentation, and business logic functionality.

Rationale:

Multitier, or n-tier, architectures are types of client/server architectures that enable an application to be accessed and executed by one or more software agents or services on the network. An N-tier architecture should be composed of layers; **graphical user interface** (**GUI**), business logic, and data should enable developing and maintaining each tier separately as technologies change. Separation of each tier may be logical or physical. Regardless of the physical system design, the structure should include well-defined boundaries between the different tiers so that changes in the system are transparent to users.
For example, N-tier architectures may employ Web services as a means of separating the presentation layer from business logic and data layers. The presentation layer serves static content through **Web pages**. A business logic layer provides dynamic content using a **J2EE application server**. Finally, a database provides the underlying information that must be shared.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Scalability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Open Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Accommodate Heterogeneity

Evaluation Criteria:

1) Test:

Does the architecture support clear boundaries between data, presentation, and business logic layers?

Procedure:

Examine the architecture for clear boundaries between data, presentation, and business logic layers.

Example:

The architecture uses Web Services to share information between the presentation and business logic layers.

BP1865

Provide sufficient program, project, or initiative metadata descriptions and automated support to enable mediation and translation of the data between interfaces.

Rationale:

Information exchanges should support known and unanticipated users. The program or project should initiate sufficient metadata descriptions and provide automated support to enable mediation and translation of data between interfaces. All of the data that can and should be shared externally beyond the programmatic bounds of your program should be defined well enough in metadata descriptions and translation of the data between interfaces should be automated.

Referenced By:

NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Policy / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Net-Centric Data Strategy (NCDS)
NESI / Part 3: Migration Guidance / Net-Centric Data Strategy (NCDS)
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Coordination of Node and Enterprise Services
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Accessibility - Operational / NCES Federated Search
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / NCES Federated Search
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Make Data Visible
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Visibility / Design Tenet: Provide Data Management
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Metadata
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - Registered / Metadata
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Metadata
NESI / Part 5: Developer Guidance / Data / Metadata

Evaluation Criteria:

1) Test:

Evaluation of interfaces and applicable mediation/translations to access that the program, project, or initiative has sufficient metadata descriptions and automated support to enable mediation and translation of the data between interfaces. Data is XML wrapped for exchange and configured to support standard transactions with headers, trailers and bodies.

Procedure:

Evaluate the degree to which data is XML wrapped for exchange and configured to support standard transactions with headers, trailers and bodies.

Evaluation of the DoD Metadata Registry entries to assess sufficient metadata descriptions and automated support the enables mediation and translation of the data between interfaces.

Example:

XML wrapped data are intend for exchange, that is configured in terms of standard transactions with headers, trailers and bodies.

BP1866

Coordinate with end users to develop interoperable materiel in support of high-value mission capability.

Rationale:

System providers acquire the materiel portion of mission capabilities that include all aspects of DOTMLP-F. An assessment by the community regarding the value of information or services provides useful direction in support of managing a mission area's portfolio of services. User feedback mechanisms provide a means of capturing and reporting user satisfaction and give portfolio managers decision-making information to steer investments, developments, and improvements. As service consumers gain access to information more quickly in the operational environment, command structures will inevitably change the manner in which IT investments are made. Service and information providers in a mission area should work together to define the processes for using the user feedback for service and information improvements because these processes are specific to a portfolio of capabilities in the Enterprise.

Referenced By:

NESI / Part 3: Migration Guidance / Migration Patterns / SOA-Enabled Migration Starting Point
NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Net-Centric Information Engineering
NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Net-Centric Information Engineering
NESI / Part 4: Node Guidance / General Responsibilities / Net-Centric Information Engineering
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Make Data Interoperable
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

Evaluation Criteria:

1) Test:

Processes exist that allow a consumer to

1. request changes in the format (syntax or semantic) of the visible data asset;
2. report a problem with a data asset;
3. request additional data from the data provider

Procedure:

Evaluation of the process a consumer would follow to

1. request changes in the format (syntax or semantic) of the visible data asset;
2. report a problem with a data asset;
3. request additional data from the data provider.

Example:

An end-to-end output management strategy, across multiple business sites and/or the enterprise.

A distributed and extensible database which make information accessible to authorized users across the enterprise.

BP1867

Use metrics to track responsiveness to user information sharing needs.

Rationale:

Information sharing metrics are defined to measure and track implementation of the net-centric approaches. Measurement techniques should be developed to ensure that metrics are captured in a useful and consistent manner. Metrics should be tagged with **DDMS**-compliant metadata and provided to the NCE to promote awareness of data management successes and areas requiring improvement.

Referenced By:

NESI / Part 4: Node Guidance / Node Computing Infrastructure / Instrumentation for Metrics
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Data / Design Tenet: Be Responsive to User Needs

Evaluation Criteria:

1) Test:

Does the program, project or initiative have metrics for determining responsiveness to user needs?

Procedure:

Evaluate the metrics being used to determine responsiveness to user data needs. If YES, describe; If NO, explain and identify a time frame for when the program, project, or initiative will have metrics for determining responsiveness to user needs; or specify NOT APPLICABLE and explain.

Example:

Examples of data metrics include percentage of Web-enabled components, progress toward service-enabling identified key functional components, and percentage of tagged community data.

BP1868

Incorporate mechanisms to enhance the survivability, resiliency, redundancy, and reliability of Computing Infrastructure (CI).

Rationale:

Computing Infrastructure (CI) must be survivable, resilient, redundant, and reliable in the presence of attacks, failures, accidents, and natural or man-made disasters. A robust CI must incorporate survivability, resiliency, redundancy, and reliability to ensure operational availability in support of information sharing in DoD, as well as externally with federal agencies, state and local governments, allies, and coalition partners. In the context of the CI, the measure of reliability is included as a critical element in ensuring high mean time between failures (MTBF).

Survivable: Survivability ensures that CI systems, subsystems, equipment, processes, procedures, or CI-related doctrine, organization, training, materiel, leadership, personnel, facilities (DOTMLPF) continue to fulfill critical mission requirements in the presence of attacks, failures, accidents, and natural or man-made disasters.

Resilient: Incorporation of resiliency into CI ensures the ability to automatically recover from, or adjust to, attacks, failures, or accidents. Fault tolerance is a key example of resilience that measures the ability to respond gracefully to an unexpected CI system, subsystem, process, or procedure failure.

Redundant: Incorporation of automatic redundancy into the CI ensures that alternative devices are available to perform the required system functionality if a primary device fails. Redundancy also ensures that system data remains accessible and corruption free when CI components fail.

Reliable: Reliable OS platforms, other software infrastructure, and hardware components are critical to ensuring that operators can depend on their ability to support system functions and applications. Bandwidth conservation mechanisms minimize latency and jitter, as well as the instability that comes from running processors and networks with high loads. Processing efficiency mechanisms, such as efficient software implementation techniques, allow applications to meet performance and latency requirements. Typically, reliability is measured in mean time between user failures (MTBUF). MTBF of CI components is one factor affecting the overall system MTBF.

A Continuity of Operations Plan (COOP) and disaster recovery planning are also key to ensuring a robust CI. The DoD Dictionary of Military Terms defines COOP as "the degree or state of being continuous in the conduct of functions, tasks, or duties necessary to accomplish a military action or mission in carrying out the national military strategy." It includes the functions and duties of the commander, as well as the supporting functions and duties performed by the staff and others acting under the authority and direction of the commander.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Availability
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Services / Design Tenet: Enterprise Service Management

Evaluation Criteria:

1) Test:

Does the program or initiative have a Continuity of Operations Plan (COOP) plan?

Procedure:

Verify existence of COOP.

Example:

Continuity of Operations Plans and Disaster Recovery Plans that include preparatory measures, response actions, and restoration activities planned or taken to ensure continuation of critical functions to maintain effectiveness, readiness, and survivability.

Technologies that allow, self-correcting mechanisms to be implemented (e.g., automatic recovery without manual intervention).

Clustering of servers, incorporation of relative addressing schemata (e.g., **DNS**), site mirroring, and provisioning of geographically distributed CI functionality are examples of fail-over implementations.

BP1875

Describe the process and protocols used to provide concurrent traffic from multiple security domains on a single IP internetwork.

Rationale:

Transport service users should implement interfaces to (or transition to) a transport infrastructure supporting fully converged IP traffic (voice, video, data, and imagery) using DoD-adopted standards (see **DISR** for appropriate standards). Transport service providers should implement converged nets as a single IP internetwork. DoD requires multiple security domains to conduct network-centric warfare.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Concurrent Transport of Information Flows
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Technical Architecture [now DISR]

Evaluation Criteria:

1) Test:

What processes and protocols are used to provide convergence of traffic (voice, video and data) from multiple security domains on a single IP internetwork?

Procedure:

Describe the process (and protocols) used to provide convergence of traffic (voice, video and data from multiple security domains on a single IP internetwork. Verify that DoD standards and products to support traffic convergence are utilized.

Example:

NSA-approved multi-level security guard.

BP1876

Provide a priority-based differentiated management of quality-of-service for traffic based on class of user, application, or mission.

Rationale:

The GIG and its components must support both QoS and CoS in accordance with the DoD QoS/CoS Roadmap and policies. The primary QoS factors that affect end-user experience include availability, throughput, delay/latency, jitter (variation in delay with time), and bit/packet loss. In addition, all GIG networks should be designed with the ability to support end-to-end treatment of multiple distinct classes of service prioritization levels. These prioritization levels require that higher-precedence data flows will be transmitted through the networks with their required QoS with greater assurance than are lower-precedence data flows. Prioritization must enforce transmission of higher-precedence data in the network, at best, concurrently with or, at worst, to the detriment of lower-precedence data flows. In the best case, sufficient resources exist to transmit data of different priorities with their required quality. Otherwise, higher-priority data must be transmitted at the expense of lower-precedence data, possibly degrading or even preempting the lower-priority data. This capability, referred to as Class of Service (CoS) support, corresponds approximately to the notion of Multi-Level Priority and Preemption (MLPP).

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Layering and Modularity
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the program, project, or initiative support a priority-based differentiated management QoS?

Procedure:

Describe the approach used to provide a priority-based differentiated management of quality-of-service.

Example:

Some applications in the GIG require firm service guarantees, while others operate correctly if they receive services that are differentiated with respect to one or more performance characteristics.
Differentiated Services or DiffServ aggregates flows into coarse classes and then treats the packets in these classes differentially. Due to this aggregation, and the resulting absence of a need to consider individual flows beyond the edges of an internet, DiffServ exhibits good scaling properties. However, in the absence of additional mechanisms, DiffServ provides only preferential, differentiated levels of service and not guarantees.

BP1877

Align end-to-end interoperable management of QoS with external networks.

Rationale:

QoS/CoS Working Group is investigating complete end-to-end QoS frameworks providing both differentiated and guaranteed QoS. They are developing a DoD roadmap and baseline architecture straw man. The architecture needs to define transport user and transport provider functions, such as where packets are labeled (application or router with Service Level Agreement).

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Does the program, project, or initiative support end-to-end interoperable management of QoS with external networks?

Procedure:

Describe the approach used to provide a priority-based differentiated management of quality-of-service across external networks.

Example:

Complete end-to-end QoS frameworks providing both differentiated and guaranteed QoS.

BP1878

Quantitative measures of QoS requirements should be supportable.

Rationale:

All GIG networks should be provisioned according to SLAs to provide QoS that meets or exceeds that required by networked applications for the transport of voice, data, video, imagery, and any other demands. The primary QoS factors that affect end-user experience include availability, throughput, delay/latency, jitter (variation in delay with time), and bit/packet loss.

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

What measures of quantitative QoS requirements are supportable, for example jitter, latency, throughput, packet loss, and others, under specific workloads?

Procedure:

Identify and describe all the QoS measurement criteria that the program, project or initiative will measure.

Example:

Jitter, latency, throughput, packet loss, etc.

BP1879

The program, project or initiative should align with the DoD QoS/CoS Working Group Roadmap.

Rationale:

Various approaches are being explored, with none yet adopted. DoD QoS/CoS Working Group is investigating complete end-to-end QoS frameworks providing both differentiated and guaranteed QoS. They are developing a DoD roadmap and baseline architecture strawman. The architecture needs to define transport user and transport provider functions, such as where packets are labeled (application or router with Service Level Agreement).

Referenced By:

NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Transport Goal
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Concurrent Transport of Information Flows
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Differentiated Management of Quality-of-Service
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Packet Switched Infrastructure

Evaluation Criteria:

1) Test:

Is the program, project, or initiative aligned with the DoD QoS/CoS Working Group roadmap?

Procedure:

Describe your program's alignment with the DoD QoS/CoS working group roadmap.

Example:

None.

BP1880

Justify, document, and obtain a waiver for all radio terminal acquisitions that are not JTRS/SCA compliant.

Rationale:

Tactical communications programs should focus on attaining the end objective of providing a family of software-programmable radios that will greatly enhance warfighters' wireless communication capabilities, while decreasing cost of ownership for infrastructure. The Joint Tactical Radio System (JTRS) will provide critical communications capabilities for the tactical wireless tails of the GIG. JTRS and its software communications architecture (SCA) continue to evolve and have become a cornerstone of the provision of future net-centric capabilities.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Software Communication Architecture
NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Communication Architecture
NESI / Part 5: Developer Guidance / Middleware / Software Communication Architecture
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Information Assurance/Security / Design Tenet: Employment of Wireless Technologies
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Concurrent Transport of Information Flows
NESI / Part 2: Traceability / ASD(NII): Net-Centric Guidance / Transport / Design Tenet: Joint Net-Centric Capabilities

Evaluation Criteria:

1) Test:

Are all of the program's, project's, or initiative's radio acquisitions JTRS/SCA compliant?

Procedure:

Describe all radio acquisitions that are not JTRS/SCA compliant.

Example:

None.

BP1881

Separate code based on required privilege.

Rationale:

Separating code based on privilege allows for each function, process, or executable to run with a minimal set of privileges.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Principle of Least Privilege
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Principle of Least Privilege
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Principle of Least Privilege

BP1888

Only enable plaintext viewing in email clients on DoD-owned and DoD-operated information systems.

Rationale:

Due to the significant risk of malicious mobile code downloaded into user workstations via email, DoD Mobile Code Policy restricts all mobile code in email independent of risk category. Disabling the automatic execution of mobile code in email is for both mobile code contained in the body of an email message and attachments. This will prevent immediate automatic execution of HTML that may download and execute mobile code from remote sites when the user clicks on a message to preview it. The user will be able to preview the message, optionally view the page source of suspicious-looking messages, and subsequently decide whether to open the attachment (the user will still be able to intentionally select the email attachment to execute HTML in that attachment.)

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Mobile Code
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Mobile Code

Evaluation Criteria:

1) Test:

Is automatic execution of all categories of mobile code in email disabled?

Procedure:

Verify that only plaintext email viewing is enabled.

Example:

BP1889

Minimize execution at elevated privilege levels to the shortest time required.

Rationale:

Holding elevated permission for a minimum time reduces the chance that a security exploit can execute arbitrary code and minimizes the impact when an exploit occurs.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Principle of Least Privilege
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Principle of Least Privilege
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Apply Principle of Least Privilege

BP1890

Compile code using the highest compiler warning level available.

Rationale:

Compiler warnings are often useful in detecting possible violations of syntax rules and mistakes introduced by developers which may lead to run time errors.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings

Evaluation Criteria:

1) Test:

Is code compiled using the highest compiler warning level available for the compiler?

Procedure:

Verify that the build script includes an applicable flag to enable the highest warning level for the compiler.

Example:

Java compilers version 5 and higher support a **`-Xlint`** compile option.

BP1891

Develop code such that it compiles without compiler warnings.

Rationale:

Compiler warnings are often useful in detecting possible violations of syntax rules and mistakes introduced by developers which may lead to run time errors.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings

BP1892

Explicitly document exceptions for valid code that produces compiler warnings.

Rationale:

It is important to document exceptions when valid code produces a compiler warning as it aids maintenance and documents the reason for the warning which is useful for future development of the code and peer reviews. Often the documentation method for a programming language will also allow for suppressing the compiler warning which prevents false positive warning in the compiler output.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Heed Compiler Warnings

BP1893

Return meaningful, but non-sensitive, information from exception handlers.

Rationale:

Purging or sanitizing exception shown to users reduces the risk of exposing information to a user that may be used to form an exploit.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Handle Exceptions

BP1898

Purchase computers which contain a Trusted Platform Module (TPM).

Rationale:

Supporting TPM is a desirable requirement at this time, since many **DoD** components want to leverage its capabilities in the future for the protection of **data at rest** (**DAR**) on mobile computing devices. TPM is readily available in the commercial marker, and in most cases is standard on new computers.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Data at Rest
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Data at Rest

BP1901

Use Universal Core (UCore) as the basis for information exchange models for systems that exchange internal data with external systems.

Rationale:

UCore defines a specification containing agreed-upon representations for the most commonly shared and universally understood concepts of "who," "what," "when" and "where." Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01E, *Interoperability and Supportability of Information Technology and National Security Systems* [R1175] recommends using UCore; this use is consistent with the *DoD Net-Centric Data Strategy*.[R1312]

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Data Interchange Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Data Management Services / Data / Data Modeling
NESI / Part 2: Traceability / DISR Service Areas / Internationalization Services / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Data Exposure Verification Tracking Sheet / Data Understandability / Data Modeling
NESI / Part 2: Traceability / Exposure Verification Tracking Sheets / Service Exposure Verification Tracking Sheet / Service Understandability - COI Data Models / Data Modeling
NESI / Part 5: Developer Guidance / Data / Data Modeling

BP1903

Include an xsd:dateTime field within long-lived XML digital signatures.

Rationale:

Just as in hand-written signatures, the time of signing in important consideration in long-lived digital signatures.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures
NESI / Part 5: Developer Guidance / Software Security / Technologies and Standards for Implementing Software Security / XML Digital Signatures

Evaluation Criteria:

1) Test:

Does the XML digital signature contain a field of type **`xsd:dateTime`**?

Procedure:

Verify the XML digital signature contains a field of type **`xsd:dateTime`**.

Example:

BP1907

Use Internet Relay Chat (IRC) bots to provide network based IRC services.

Rationale:

Internet Relay Chat (IRC) bots are stand-alone, independent programs, that connect to IRC Servers as clients. IRC bots commonly provide services in an IRC system; for example, keeping chat channels open, protecting chat channels, and recording messages for users who are currently off-line.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Text Conferencing
NESI / Part 2: Traceability / DISR Service Areas / Environment Management / Services / Core Enterprise Services (CES) / Collaboration Services / Text Conferencing
NESI / Part 4: Node Guidance / Services / Core Enterprise Services (CES) / Collaboration Services / Text Conferencing

BP1908

Provide bidirectional mediation between transport protocols mandated in the Defense IT Standards Registry (DISR) when implementing an Enterprise Service Bus (ESB).

Rationale:

ESBs provide transport protocol agnostic messaging between service producers and consumers. ESBs use transport protocol mediation to achieve this goal. Service interactions are not simple, one-way activities, but require an interactive dialog between the service producer and the consumer. To achieve this dialog, all protocol mediation needs to be bi-directional. Suporting mediation for transport protocols specified by the DISR allows message proucers and consumers flexibility in choice of transport protocol.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Enterprise Service Bus (ESB)
NESI / Part 5: Developer Guidance / Middleware / Enterprise Service Bus (ESB)

BP1909

Provide for filtering of XML messages using XML Path Language (XPath) when implementing an Enterprise Service Bus (ESB).

Rationale:

ESBs provide filtering and restricting of messages in order to match messge producers and consumers. XPath is a language specifically intended for effectively and efficiently finding information within an XML document. Therefore, syntax and tools that are based on XPath are preferred filter methods for messages formulated as XML documents sent over an ESB.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Enterprise Service Bus (ESB)
NESI / Part 5: Developer Guidance / Middleware / Enterprise Service Bus (ESB)

BP1911

Provide for routing of messages based on message content when implementing an Enterprise Service Bus (ESB).

Rationale:

The ability to route messages based on message content allows for flexible dynamic matching of content producers and consumers.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Enterprise Service Bus (ESB)
NESI / Part 5: Developer Guidance / Middleware / Enterprise Service Bus (ESB)

BP1913

Provide for mediation between synchronous and asynchronous messages when implementing an Enterprise Service Bus (ESB).

Rationale:

ESBs support synchronous and asynchronous communication models. Allowing for mediation between synchronous consumers and asynchronous producers, and vice versa, allows for more flexible matching of message producers and consumers.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Distributed Computing Services / Enterprise Service Bus (ESB)
NESI / Part 5: Developer Guidance / Middleware / Enterprise Service Bus (ESB)

BP1922

Design systems to have security as a core capability.

Rationale:

Adding non-functional capabilities, such as timeliness, fault management, and security, to a designed or implemented system usually is not cost-effective, if possible to do at all. Those capabilities are integral to the operation and thus significantly affect the design and implementation from the beginning of the initial modeling.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Operating System Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Practice Defense in Depth
NESI / Part 2: Traceability / DISR Service Areas / Security Services / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Practice Defense in Depth
NESI / Part 5: Developer Guidance / Software Security / Policies and Processes for Implementing Software Security / Secure Coding and Implementation Practices / Practice Defense in Depth
NESI / Part 4: Node Guidance / Node Computing Infrastructure / Time-Critical Operations

BP1923

Employ an operating system that supports simultaneously IPv4 and IPv6.

Rationale:

In order to support applications that require both IPv4 and IPv6 communications, the operating system must also support both IPv4 and IPv6 simultaneously.

Referenced By:

NESI / Part 2: Traceability / DISR Service Areas / Communications Applications / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition
NESI / Part 4: Node Guidance / Node Transport / Network Layer / Internet Protocol (IP) / IPv4 to IPv6 Transition

Evaluation Criteria:

1) Test:

Does the operating system support dual stack IPv4 and IPv6?

Procedure:

Check the operating system's IP configuration for dual IPv4 and IPv6 configurations.

Example:

None

Glossary

.NET Framework		The .NET Framework is an integral Windows component that supports building and running the next generation of applications and XML Web services. The .NET Framework has two main components: the common language runtime and the .NET Framework class library. (Source: MSDN ***.NET Framework Conceptual Overview***, http://msdn.microsoft.com/en-us/library/zw4w595w.aspx)
Access Control		Limiting access to information system resources only to authorized users, programs, processes, or other systems. (Source: *National Information Assurance (IA) Glossary*, CNSSI 4009, revised June 2006) ***Note:*** *See also the following:* • ***Access Control List*** *(**ACL**) [GL1889]* • ***Discretionary Access Control*** *(**DAC**) [GL1197]* • ***Role-Based Access Control*** *(**RBAC**) [GL1643]*
Access Control List	ACL	In computer security, ACL is a concept used to enforce privilege separation. It is a means of determining the appropriate access rights to a given object depending on certain aspects of the process that is making the request, principally the process's user identity. In networking, ACL refers to a list of ports and services that are available on a host, each with a list of hosts and/or networks permitted to use the service. Both individual servers as well as routers can have access lists. Access lists are used to control both inbound and outbound traffic, and in this context they are similar to firewalls. (Source: http://en.wikipedia.org/wiki/Access_control_list)
Accredited Standards Committee X12	ASC X12	In 1979, the **American National Standards Institute** (**ANSI**) chartered the Accredited Standards Committee (ASC) X12 to develop uniform standards for interindustry electronic exchange of business transactions-electronic data interchange (**EDI**). (Source: http://www.x12.org/x12org/about/faqs.cfm#b1)
Active Directory	AD	An implementation of Lightweight Directory Access Protocol (LDAP) directory services by Microsoft for use in Windows environments; allows administrators to assign enterprise-wide policies, deploy programs to many computers, and apply critical updates to an entire organization. An Active Directory stores information and settings relating to an organization in a central, organized, accessible database. Active Directory networks can vary from a small installation with a few hundred objects, to a large installation with millions of objects. (Source: http://en.wikipedia.org/wiki/Active_Directory)

Active Server Page	ASP	A script that is executed by Microsoft Internet Information Services. The output is returned to the user as **HTML**. Typically, an ASP script generates a customized Web page on the fly before sending it to the user. ASPs are specific to Microsoft, only run on **IIS** or **PWS**, can contain HTML, **JScript**, and **VBScript**, and can access **COM** components.
ActiveX		An ActiveX control is similar to a Java **applet**. However, ActiveX controls have full access to the Windows OS. This gives them much more power than Java applets, plus a risk that the applet may damage software or data on your machine. To control this risk, Microsoft developed a registration system so that browsers can identify and authenticate an ActiveX control before downloading it. Another difference between Java applets and ActiveX controls is that Java applets can be written to run on all platforms, whereas ActiveX controls are currently limited to Windows environments.
Adapter		An intermediary that translates between incompatible components interfaces, allowing them to communicate.
Aggregation		When information is derived from multiple sources a mediator service may aggregate the data and thus make many services appear to be one. I1148 Note: See **Mediation**.
All Views	AV	The DoDAF All-Views (AV) products provide information pertinent to the entire architecture but do not represent a distinct view of the architecture. AV products set the scope and context of the architecture. The scope includes the subject area and timeframe for the architecture. The setting in which the architecture exists comprises the interrelated conditions that compose the context for the architecture. These conditions include doctrine; tactics, techniques, and procedures; relevant goals and vision statements; concepts of operations; scenarios; and environmental conditions. (Source: *DoDAF* v1.5 Volume 1: Definitions and Guidelines, 23 April 2007)
American National Standards Institute	ANSI	Administrator and coordinator of the United States private-sector voluntary standardization system. ANSI facilitates the development of American National Standards (ANS) by accrediting the procedures of standards-developing organizations. The Institute remains a private, nonprofit membership organization supported by a diverse constituency of private and public sector organizations. (Source: http://web.ansi.org/)

American Standard Code for Information Interchange	ASCII	ASCII is a character set and a character encoding based on the Roman alphabet as used in modern English (see English alphabet). ASCII codes represent text in computers, in other communications equipment, and in control devices that work with text. Most often, nowadays, character encoding has an ASCII-like base.
		
		ASCII defines the following printable characters, presented here in numerical order of their ASCII value:
		
		`!"#$%'()*+,-./0123456789:; ?`
		`@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
		`` `abcdefghijklmnopqrstuvwxyz{|}~( ``
		
		(Source: http://en.wikipedia.org/wiki/ASCII)

Applet		A J2EE component that typically executes in a Web browser. Applets can also execute in a variety of other applications or devices that support the applet programming model. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Application		An application is a software program that performs a specific function directly for a user, with or without requiring extraordinary authority or privileges such as system-level control and monitoring, administrative or "super user" rights, or root-level access. (Source: derived from Committee on National Security Systems Instruction 4009, *National Information Assurance Glossary* [R1339])
Application Environment Profile	AEP	The AEP describes the exact functionality supported by the Operating Environment of the **SCA** specification.
Application Programming Interface	API	A special type of interface that specifies the calling conventions with which one component may access the resources and services provided by another component. APIs are defined by sets of procedures or function-invocation specifications. An API is a special case of an interface.
Application Server		A platform for developing and deploying multi-tier distributed enterprise applications.
Architectural Style		An architectural style is the combination of distinctive features in which **architecture** is performed or expressed. (Source: http://www.opengroup.org/projects/soa/doc.tpl?gdid=10632)
Architecture		(1) The structure of components, their relationships, and the principles and guidelines governing their design and evolution over time. (2) A high-level design that provides decisions about the problem(s) that the product will solve, component descriptions, relationships between components, and dynamic operation description. (3) A framework or structure that portrays relationships among all the elements of the subject force, system, or activity. Also, the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution. The organizational structure of a system or component, their

		relationships, and the principles and guidelines governing their design and evolution over time. (Source: IEEE Std 610.12)
Assistant Secretary of Defense for Networks and Information Integration	ASD (NII)	(Source: http://www.dod.mil/nii/)
Asymmetric Key Cryptography		Synonym for **Public Key Cryptography**.
Attribute		A distinct characteristic of an object. Real-world object attributes are often specified in terms of their physical traits, such as size, shape, weight, and color. Cyberspace object attributes might describe size, type of encoding, and network address. (Source: ***Web Services for Remote Portlets Specification***, ***Appendix A: Glossary***; http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf)
Authentication		The process that verifies the identity of a user, device, or other entity in a computer system, usually as a prerequisite to allowing access to resources in a system. The Java servlet specification requires three types of authentication (basic, form-based, and mutual) and supports digest authentication. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Authorization		The process by which access to a method or resource is determined. Authorization depends on the determination of whether the principal associated with a request through authentication is in a given security role. A security role is a logical grouping of users defined by the person who assembles the application. A deployer maps security roles to security identities. Security identities may be principals or groups in the operational environment. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Basic Object Adapter	BOA	The Basic Object Adapter was an early (v1) CORBA component; see the **Portable Object Adapter** (**POA**).
Binary XML		Binary XML is a format which does not conform to the XML specification yet maintains a well-defined, useful [i.e., practical systems may take advantage of this relationship with little additional effort] relationship with XML. (Source: derived from Section 2.1 ***Definition of Binary XML*** in the ***XML Binary Characterization W3C Working Group Note***, 31 March 2005; http://www.w3.org/TR/xbc-characterization/)
Browser		Short for **Web browser**, a software application used to locate and display Web pages. (Source: http://www.webopedia.com/TERM/b/browser.html)
Business Logic		The code that implements the functionality of an application. In the Enterprise JavaBeans architecture, this logic is implemented by the methods of an enterprise bean. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)

Business Process Execution Language	BPEL	BPEL is emerging as the standard for assembling a set of discrete services into an end-to-end process flow, radically reducing the cost and complexity of process integration initiatives. (Source: http://www.oracle.com/technology/products/ias/bpel/index.html)
Canonicalization		The process of converting data that has more than one possible representation into a "standard" canonical representation. This can be done to compare different representations for equivalence, to count the number of distinct data structure , to improve the efficiency of various algorithms by eliminating repeated calculations, or to make it possible to impose a meaningful sorting order. (Source: http://en.wikipedia.org/wiki/Canonicalization) When referring to XML, the process of converting an XML document to a form that is consistent to all parties. Used when signing documents and interpreting signatures. Any XML document is part of a set of XML documents that are logically equivalent within an application context. Generally, if two documents have the same canonical form, then the two documents are logically equivalent within the given application context. Methods exist for generating a physical representation, the canonical form, of an XML document that accounts for the permissible changes. Note that two documents may have differing canonical forms yet still be equivalent in a given context based on application-specific equivalence rules for which no generalized XML specification could account.
Capability Development Document	CDD	Provides operational performance attributes, including supportability, for the acquisition community to design the proposed system. Includes key performance parameters (KPP) and other parameters that guide the development, demonstration, and testing of the current increment. Outlines the overall strategy for developing full capability. (Source: http://www.dau.mil/pubs/glossary/12th_Glossary_2005.pdf)
Capability Production Document	CPD	Addresses the production attributes and quantities specific to a single increment of an acquisition program. Supersedes threshold and objective performance values of the CDD. (Source: http://www.dau.mil/pubs/glossary/12th_Glossary_2005.pdf)
Cascading Style Sheet	CSS	Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents. (Source: http://www.w3.org/Style/CSS/)
Certificate	CERT	A certificate which uses a digital signature to bind together a public key with an identity information such as the name of a person or an organization, their address, and so forth. The certificate can be used to verify that a public key belongs to an individual. (Source: http://en.wikipedia.org/wiki/Certificate_%28cryptography%29)
Certificate Authority	CA	A trusted organization which issues digital public key certificates for use by other parties. It is an example of a trusted third party. CAs are characteristic of many public key infrastructure (PKI) schemes. (Source: http://en.wikipedia.org/wiki/Certificate_authority)

Certificate Revocation List	CRL	A list of certificates (more accurately, their serial numbers) which have been revoked, are no longer valid, and should not be relied upon by any system user. (Source: http://en.wikipedia.org/wiki/Certificate_Revocation_List)
Check Constraint		A constraint based on a user-defined condition - generally documented in a database domain - that has to evaluate to true for the contents of a data base column to be valid.
Chief Information Officer	CIO	Job title for a manager responsible for **Information Technology** (IT) within an organization; often reports to the chief executive officer or chief financial officer. For information on the Assistant Secretary of Defense for Networks and Information Integration (ASD/NII)/DoD CIO see DoDD 5144.1 of 2 May 2005. (Source: http://en.wikipedia.org/wiki/Chief_Information_Officer)
Cipher Text	CT	Data that has been **encrypted**. Cipher text is unreadable until it has been converted into Plain Text (PT) (decrypted) with a key. (Source: http://www.webopedia.com/TERM/C/cipher_text.html)
Client		A system entity that accesses a Web service. (Source: http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf)
Client-Certificate Authentication		An authentication mechanism that uses HTTP over SSL, in which the server and (optionally) the client authenticate each other with a public key certificate that conforms to a standard that is defined by X.509 Public Key Infrastructure. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Cohesion		The manner and degree to which the tasks performed by a single software module are related to one another. Types include coincidental, communicational, functional, logical, procedural, sequential, and temporal. Synonym: module strength. Contrast with **coupling**. In a well-designed, highly modular software design, the modules will have high cohesion; that is, each will have a clearly defined set of functions that have a close relationship to each other. This facilitates changes to modules since the changes will affect only the closely-related functions. In contrast, modules that contain multiple, unrelated functions blur the integrity of the software's design since the unrelated functions are bound into a single module, thereby creating dependencies that inhibit the ability to easily make changes. (Source: IEEE Std 610.12-1990)
Collaboration		Portal members can communicate synchronously through chat or messaging, or asynchronously through threaded discussion, blogs, and email digests (forums).
Collaboration Management Office	CMO	DISA organization responsible for fielding, sustaining and managing the life cycle of the **Defense Collaboration Tool Suite** (DCTS).
Command, Control, Communications,	C4ISR	

Computers, and Intelligence, Surveillance, and Reconnaissance		
Command and Control	C2	(DoD) The exercise of authority and direction by a properly designated commander over assigned and attached forces in the accomplishment of the mission. Command and control functions are performed through an arrangement of personnel, equipment, communications, facilities, and procedures employed by a commander in planning, directing, coordinating, and controlling forces and operations in the accomplishment of the mission. (Source: DoD, ***Department of Defense Dictionary of Military and Associated Terms***, JP 1-02, 12 April 2001 as amended through 17 October 2008)
Commercial Off-The-Shelf	COTS	A term for systems that are manufactured commercially, and may be tailored for specific uses. (Source: http://en.wikipedia.org/wiki/Commercial_off-the-shelf)
Common Access Card	CAC	A DoD-wide smart card used as the identification card for active duty Uniformed Services personnel (to include the Selected Reserve), DoD civilian employees, eligible contractor personnel, and eligible foreign nationals; the primary platform for the Public Key Infrastructure (PKI) authentication token used to access DoD computer networks and systems in the unclassified environment and, where authorized by governing security directives, the classified environment; and the principal card enabling physical access to buildings, facilities, installations, and controlled spaces as described in DoD Directive 8190.3, "Smart Card Technology," 31 August 2002. ***Note:*** *The Defense Manpower Data Center (DMDC) Common Access Card site (http://www.dmdc.osd.mil/smartcard) contains additional information, reports and developer support concerning the DoD CAC implementation.* (Source: DoD Instruction 8520.2, 1 April 2004, [R1206] Enclosure (2) Definitions, page 13)
Common Business Oriented Language	COBOL	COBOL is a third-generation programming language. Its name is an acronym, for COmmon Business Oriented Language, defining its primary domain in business, finance, and administrative systems for companies and governments. (Source: http://en.wikipedia.org/wiki/COBOL)
Common Gateway Interface Script	CGI Script	CGI is a standard for interfacing external applications with information servers, such as HTTP or Web servers. A plain HTML document that the Web daemon retrieves is static, which means it exists in a constant state: a text file that doesn't change. A CGI program, on the other hand, is executed in real time, so it can output dynamic information.
Common Language Runtime	CLR	CLR, at the very core of the **.NET** Framework, encapsulates all the services used from the operating system by compilers of higher level languages such as Visual Basic .NET, Visual C++ .NET, Visual J# .NET and Visual C# .NET. The higher level languages ultimately are translated into native code that directly accesses the CLR.

Common Object Request Broker Architecture	CORBA	CORBA "wraps" code written in another language into a bundle containing additional information on the capabilities of the code inside, and explaining how to call it. The resulting wrapped objects can then be called from other programs (or CORBA objects) over the network. The CORBA specification defines APIs, communication protocol, and object/ service information models to enable heterogeneous applications written in various languages running on various platforms to interoperate. (Source: http://en.wikipedia.org/wiki/CORBA)
Community of Interest	COI	A COI is a collaborative group of users that must exchange information in pursuit of its shared goals, interests, missions, or business processes and therefore must have shared vocabulary for the information it exchanges. (Source: DoDD 8320.02, 2 December 2004, *Data Sharing in a Net-Centric Department of Defense*)
Community of Interest Service		A service that may be offered to the enterprise, but is owned and operated by a **Community of Interest** to provide or support a well-defined set of mission functions and associated information.
Compiler		A computer program that translates programs expressed in a high-order language into their machine language equivalent. (Source: IEEE Std 610.12-1990)
Complex Data		Complex data can be represented in a complex data structure or can be mapped into a relational or flat structure with additional metadata provided to represent the complex relationships.
Complex Semi-Structured Data		Complex Semi-Structured Data has partial metadata. It includes data defined in COBOL copybooks and Electronic Data Interchange standards ANSI X.12 and Health Level 7 (HL7). Semi-structured data can be as complex or more so as any Complex Structured data. It can map into or be XML. It may also be missing some metadata or an XSD.
Complex Structured Data		Complex Structured Data has well-defined metadata. It includes data represented in XML documents with deeply hierarchical and recursive structures. Complex data can be represented in a complex data structure or can be mapped into a relational or flat structure with additional metadata provided to represent the complex relationships. Although complex structured data is generically a property of object oriented databases, the Complex Data Structures can be filled from any source.
Complex Unstructured Data		Complex Unstructured Data has little or no metadata. It includes data in binary files, spreadsheets, documents, and print streams.
Component		One of the parts that make up a system. A component may be hardware or software and may be subdivided into other components. Note the terms ***module***, ***component***, and ***unit*** are often used interchangeably or defined to be sub-elements of one another in different ways depending on the context. The relationship of these terms is not yet standardized. (Source: IEEE Std 610.12-1990)

		Note: *See* ***system component*** *and* ***software component****.*
Component-Based Software		Mission applications that are architected as components integrated within a component framework.
Component Object Model	COM	A Microsoft software architecture for building component-based applications. COM objects are discrete components, each with a unique identity, which expose interfaces that allow applications and other components to access their features. COM objects are more versatile than Win32 DLLs because they are completely language-independent, have built-in inter-process communications capability, and easily fit into an object-oriented program design. COM was first released in 1993 with OLE2, largely to replace the inter-process communication mechanism DDE used by the initial release of OLE. **ActiveX** is based on COM.
Computer Network Defense	CND	Defensive measures to protect and defend information, computers, and networks from disruption, denial, degradation, or destruction. (Source: DoD, Department of Defense Dictionary of Military and Associated Terms, JP 1-02, 12 April 2001 as amended through 17 October 2008)
Computer Network Defense Service Provider	CNDSP	Those organizations responsible for delivering protection, detection and response services to its users. CNDS providers must provide for the coordination service support of a CNDS/CA. CNDS is commonly provided by a Computer Emergency or Incident Response Team (CERT/CIRT) and may be associated with a Network Operations (NetOps) and Security Center (NOSC). (Source: DoD Directive O-8530.1, ***Computer Network Defense (CND)***,[R1191] 8 January 2001, Enclosure 2 Definitions, p. 12)
Conceptual Model		Captures the concepts of the relational database and can help enforce the first three normalization rules.
Condition		A variable of the operational environment or situation in which a unit, system, or individual is expected to operate that may affect performance. A **DDS** Condition is attached to a **WaitSet** and indicates which condition the application is waiting for asynchronously: **`StatusCondition`**, **`ReadCondition`** or **`QueryCondition`**.
Confidentiality		The property that data is not made available to unauthorized individuals, entities, or processes.
Configuration Control Board	CCB	Also Change Control Board. Duties include reviewing change requests, making decisions, and communicating decisions made to affected groups and individuals. Represents the interests of program and project management by ensuring that a structured process is used to consider proposed changes and incorporate them into a specified release of a product.

Container		An entity that provides life-cycle management, security, deployment, and runtime services to J2EE components. Each type of container (EJB, Web, JSP, servlet, applet, and application client) also provides component-specific services. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Content Discovery Service	CDS	**Net-Centric Enterprise Services** (**NCES**) service that provided a Federated Search capability.
Core Enterprise Services	CES	Core Enterprise Services (CES) are a small set of **services** provided by the Enterprise Information Environment Mission Area (EIEMA). Some of the CES services will be centrally provided on behalf of the DoD while others might involve local provisioning. For locally provisioned services, EIEMA provides guidance to ensure consistent implementation throughout the DoD. (Source: *DoD Net-Centric Services Strategy*, Section 3.1 [R1313])
Coupling		The manner and degree of interdependence between software modules. Types include common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling, and pathological coupling. Contrast with **cohesion**. In a well-designed, highly modular software design, the coupling between modules will be minimized. This facilitates changing and replacing modules with minimal effect on other modules within the system. (Source: IEEE Std 610.12-1990)
Credentials		The information describing the security attributes of a principal. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
CRL Distribution Point	CDP	The location where the **Certificate Authority** (**CA**) puts the **Certificate Revocation List** (**CRL**) for relying parties to obtain the most current CRL.
Customized Delivery		Smart push-and-pull of data reduces overload and provides the requested data to operators when they need it. Tailored discovery, publish, and subscribe capabilities allow operators to register for specific data and services in specific timeframes.
Data		Unprocessed information; information without context.
Data Architect		A Data Architect is a job title associated with a person within an organization responsible for making sure the organization's strategic goals are optimized through the use of enterprise data standards. This frequently involves creating and maintaining a centralized registry of **metadata**. Data Architecture includes topics such as metadata management, business semantics, data modeling and metadata workflow management. A Data Architect's job frequently includes the set up a **metadata registry** to allow domain-specific stakeholders to maintain their own **data elements**.

Data Asset		Any entity that is composed of data. For example, a database is a data asset that contains data records (e.g., system or application output files, databases, documents, or Web pages). The term data asset also refers to services that provide access to data. For example, a service that returns individual records from a database is considered a data asset since it deals mainly in the function of providing data. Similarly, a Web site that returns data in response to specific queries (e.g., www.defenselink.mil) is considered a data asset. (Source: DoD Net-Centric Data Strategy, 9 May 2003 [R1172])
Data at Rest	DAR	Data at Rest refers to all data in computer storage (e.g., on hard disk drives, CDs/DVDs, floppy disks, thumb drives, PDAs, cell phones, other removable storage media, etc.) while excluding data that is traversing a network (data in transit) or temporarily residing in computer memory to be read or updated (data in use). Source: DoD Policy Memorandum, Encryption of Sensitive Unclassified Data at Rest on Mobile Computing Devices and Removable Storage Media R1330: DoD Memorandum , **Encryption of Sensitive Unclassified Data at Rest on Mobile Computing Devices and Removable Storage Media** Chief Information Officer . [http://iase.disa.mil/policy-guidance/dod-dar-tpm-decree07-03-07.pdf]
Database Data		Data stored in database columns in database tables in a relational database. The set of data records which a relational database is populated. Generally understood to refer to application data and not metadata.
Database Management System	DBMS	A system, usually automated and computerized, for managing any collection of compatible, and ideally normalized, data. (Source: http://en.wikipedia.org/wiki/DBMS)
Data Categorization		
Data-Centric		An approach for the design and implementation of systems, applications, services or software that emphasis the data rather than the operations. It implies that the data is physically separated from the code and consequently can be maintained independently (loose coupling between code and data).
Data-Centric Publish-Subscribe	DCPS	The Data-Centric Publish-Subscribe is a lower level layer of the **DDS** infrastructure that is targeted towards the efficient delivery of the proper information to the proper recipients.
Data Dictionary		A data dictionary is set of metadata that contains definitions and representations of **data elements**. Within the context of a DBMS, a data dictionary is a read-only set of tables and views. The data dictionary may be considered a database in its own right.

Data Distribution Service for Real-Time Systems	DDS	DDS is a recently-adopted OMG standard that is the first open international middleware standard directly addressing publish-subscribe communications for real-time and embedded systems. DDS introduces a virtual Global Data Space where applications can share information by simply reading and writing data-objects addressed by means of an application-defined name (Topic) and a key. DDS features fine and extensive control of QoS parameters, including reliability, bandwidth, delivery deadlines, and resource limits. DDS also supports the construction of local object models on top of the Global Data Space. (Source: OMG Data Distribution Portal, http://portals.omg.org/dds)
Data Element		A data element is an atomic unit of data that has the following: • an identification such as a data element name • a clear data element definition • one or more representation terms • optional enumerated values
Data Element Gallery		The Data Element Gallery is an important component of the Metadata Registry and Clearinghouse. The Data Element Gallery provides its users with access to **data elements** that are commonly used by the Department of Defense such as country codes and U.S. state codes. Users may search the registry, compare data elements, and download an Access database containing the available elements. See the DoD Metadata Registry, http://metadata.dod.mil.
Data Exposure		The steps necessary to set up the metadata infrastructure associated with a net-centric data strategy.
Data Local Reconstruction Layer	DLRL	The Data Local Reconstruction Layer is an optional part of the **DDS** specification that provides a higher level layer allowing for a simpler integration of the DDS into the application layer.
Data Modeling	DM	Modeling is an essential step in understanding the data that will comprise a system. The end products of data modeling can be XML schemas or RDBMS schema definitions. Many COIs create their own data models, such as the Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM) data model for the **C2** community.
Data Publishing		The steps necessary to make data available within the net-centric data strategy infrastructure.
Data Structure		In computer science, a data structure is a way of storing data in a computer so that it can be used efficiently. Often a carefully chosen data structure will allow a more efficient algorithm to be used. The choice of the data structure often begins from the choice of an abstract data structure. A well-designed data structure allows a variety of critical operations to be performed, using as few resources, both execution time and memory space, as possible. Data structures are implemented using the data types, references and operations on them provided

		by a programming language. (Source: http://en.wikipedia.org/wiki/Data_structure)
Data Type		A data type is a constraint placed upon the interpretation of data in a type system in computer programming. Common types of data in programming languages include primitive types (such as integers, floating point numbers or characters), tuples, records, algebraic data types, abstract data types, reference types, classes and function types. A data type describes representation, interpretation and structure of values manipulated by algorithms or objects stored in computer memory or other storage device. The type system uses data type information to check correctness of computer programs that access or manipulate the data. (Source: http://en.wikipedia.org/wiki/Data_type)
DDS DataReader		The **DDS** **`DataReader`** acts as a typed (i.e., dedicated to only one application data type) accessor to a subscriber. The **`DataReader`** class allows the application to declare the data it wishes to receive (i.e., make a subscription) and access the data received by the attached **Subscriber**.
DDS DataWriter		A **DDS** **`DataWriter`** acts as a typed (i.e., dedicated to only one application data type) accessor to a publisher. The **`DataWriter`** class allows the application to set the value of the data to be published under a given **Topic**.
DDS DomainParticipant		A **DDS** domain participant represents the local membership of the computer process in a **domain**. A domain is a distributed concept that links all the computer processes able to communicate with each other. It represents a communication plane; only the **publishers** and the **subscribers** attached to the same domain may interact. A computer process can run on the behalf of some user or application.
DDS Global Data Space		Underlying any **data-centric** publish subscribe system is a data model. In **DDS**, this model defines the global data space and specifies how **Publishers** and **Subscribers** refer to portions of this space. (See DDS **Domain**)
DDS Listener		A **DDS** **`Listener`** is used to provide a callback for synchronous access. Listeners provide a generic mechanism for the middleware to notify the application of relevant asynchronous events, such as arrival of data corresponding to a subscription, violation of a **QoS** setting, etc. Each **DCPS** entity supports its own specialized kind of listener. **`Listener`** operations are invoked using a middleware-provided thread.
DDS Publication		A **DDS** publication is defined by the association of a **DataWriter** to a **publisher**. This association expresses the intent of the application to publish the data described by the DataWriter in the context provided by the publisher.
DDS Publisher		A **DDS** publisher is an object responsible for data distribution. It may publish data of different data types. The **DataWriter** is the object the application must use to communicate to a publisher the existence and value of data-objects of a given type. When data-object values have been communicated to the publisher through the appropriate **`DataWriter`**, it is the publisher's responsibility to perform the

		distribution (the publisher will do this according to its own QoS, or the **QoS** attached to the corresponding **`DataWriter`**).
DDS Subscriber		A **DDS** subscriber is an object responsible for receiving published data and making it available (according to the Subscriber's **QoS**) to the receiving application. It may receive and dispatch data of different specified types. To access the received data, the application must use a typed **DataReader** attached to the subscriber.
DDS Subscriber Access API		**DDS** defines two **APIs** that provide subscriber access: **Listeners** and the dual **Condition/WaitSet** infrastructure allow applications to be notified when changes occur in a **DCPS** communication.
DDS Subscription		A **DDS** subscription is defined by the association of a **DataReader** with a **subscriber**. This association expresses the intent of the application to subscribe the data described by the **`DataReader`** in the context provided by the subscriber.
DDS WaitSet		A **DDS** **`WaitSet`** associated with one or several **Condition** objects provides asynchronous data access. **`WaitSets`** and their associated **`Conditions`** provide the means for an application thread to block waiting for the same events that can be received via a **Listener**. Using a **`WaitSet`** the application can handle the event in its own thread instead of the middleware provided thread used for **`Listeners`**.
Defense Acquisition University	DAU	The mission of the DAU is to provide practitioner training, career management, and services to enable the DoD Acquisition, Technology and Logistics (AT&L) community to make smart business decisions and deliver timely and affordable capabilities to the warfighter. (Source: http://www.dau.mil/about-dau/docs/mission_vision.ppt)
Defense Collaboration Tool Suite	DCTS	A flexible, integrated set of applications providing interoperable, synchronous, and asynchronous collaboration capability to the Department of Defense Agencies, Combatant Commands, and Military Services.
Defense Information System Network	DISN	The Defense Information System Network (DISN) has been the Department of Defense's enterprise network for providing data, video and voice services for more than 40 years. (Source: http://www.disa.mil/main/support/dss.html)
Defense Information Systems Agency	DISA	Combat support agency responsible for planning, engineering, acquiring, fielding, and supporting global net-centric solutions to serve the needs of the President, Vice President, the Secretary of Defense, and other DoD Components, under all conditions of peace and war. (Source: http://www.disa.mil/main/about/missman.html)
Defense IT Standards Registry	DISR	The DoD IT Standards Registry (DISR) is an online repository (http://disronline.disa.mil) for a minimal set of primarily commercial IT standards formerly captured in the Joint Technical Architecture (JTA), Version 6.0. These standards are used as the "building codes" for all systems being procured in the Department of Defense. Use of these building codes facilitates interoperability among systems and integration of new systems into the Global Information Grid (GIG). In addition, the

		DISR provides the capability to build profiles of standards that programs will use to deliver net-centric capabilities. (Source: http://akss.dau.mil/dag/GuideBook/IG_c7.2.4.2.asp)
Department of Defense	DoD	The Department of Defense is America's oldest and largest government agency. The DoD mission is to provide the military forces needed to deter war and to protect the security of the United States. (Source: adapted from *DoD 101, An Introductory Overview of the Department of Defense*; http://www.defenselink.mil/pubs/dod101/; accessed 30 April 2009)
Deployment Descriptor		An XML file provided with each module and J2EE application that describes how they should be deployed. The deployment descriptor directs a deployment tool to deploy a module or application with specific container options and describes specific configuration requirements that a deployer must resolve. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Deprecate		Deprecation is the gradual phasing-out of features such as guidance, software or programming language features. Guidance, features or methods marked as deprecated are considered obsolete, and further use is discouraged. The guidance features or methods are still valid although error messages as warnings may occur when they are referenced. These serve to alert the user to the fact that the feature may be removed in future releases. Features get marked as deprecated, rather than simply removed, in order to provide backward compatibility end users.
Deserialization		Deserialization is the reverse process of **serialization**. A stream of data is converted back into a complex object. ***Note:*** *The process of transferring data using serialization and deserialization is called* ***marshalling****.*
Design Pattern		General repeatable solution to a commonly-occurring problem in software design. A design pattern isn't a finished design that can be transformed directly into code; it is a description or template for how to solve a problem that can be used in many different situations. (Source: http://en.wikipedia.org/wiki/Design_pattern_%28computer_science%29)
Digest		A cryptographic checksum of an octet stream.
Digital Signature		A value computed with a cryptographic algorithm and bound to data in such a way that intended recipients of the data can use the signature to verify that the data has not been altered and/or has originated from the signer of the message, providing message integrity and authentication. The signature can be computed and verified with symmetric key algorithms, where the same key is used for signing and verifying, or with asymmetric key algorithms, where different keys are used for signing and verifying (a private and public key pair are used).

Digital Signature Algorithm	DSA	The Digital Signature Algorithm (DSA) is a United States Federal Government standard for digital signatures. It was proposed by the National Institute of Standards and Technology (NIST) in August 1991 for use in their Digital Signature Standard (DSS), specified in **Federal Information Processing Standard** (**FIPS**) 186, adopted in 1993. A minor revision was issued in 1996 as FIPS 186-1, and the standard was expanded further in 2000 as FIPS 186-2. (Source: http://en.wikipedia.org/wiki/Digital_Signature_Algorithm)
Directory Service		A directory service organizes computerized content and runs on a directory server computer. It is not to be confused with the directory itself, which is the database that holds the information about objects that are to be managed by the directory service. The directory service is the interface to the directory and provides access to the data that is contained in that directory. It acts as a central authority that can securely authenticate resources and manage identities and relationships between them. (Source: http://en.wikipedia.org/wiki/Directory_service)
Discretionary Access Control	DAC	Means of restricting access to objects based on the identity and need-to-know of users and/or groups to which the object belongs. Controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (directly or indirectly) to any other subject. (Source: National Information Assurance (IA) Glossary, CNSSI 4009, revised June 2006)
Distributed Component Object Model	DCOM	Distributed Component Object Model (DCOM) is a Microsoft proprietary technology for software components distributed across several networked computers to communicate with each other. It extends Microsoft's COM, and provides the communication substrate under Microsoft's COM+ application server infrastructure. It has been deprecated in favor of Microsoft **.NET**.
Doctrine, Organization, Training, Materiel, Leadership, Personnel, Facilities	DOTMLPF	
Document Object Model	DOM	An API for accessing and manipulating XML documents as tree structures. DOM provides platform-neutral, language-neutral interfaces that enable programs and scripts to dynamically access and modify content and structure in XML documents. (Source: *J2EE 1.4 Glossary*, http://java.sun.com/j2ee/1.4/docs/glossary.html)
Document Type Definition	DTD	An optional part of the XML document prolog, as specified by the XML standard. The DTD specifies constraints on the tags and tag sequences that can be in the document. The DTD has a number of shortcomings, however, and this has led to various schema proposals. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
DoD Architecture Framework	DoDAF	The DoD Architecture Framework (DoDAF) Version 2.0 is the prescribed framework for all Department architectures, and represents a substantial shift in approach. It places emphasis upon a disciplined process of

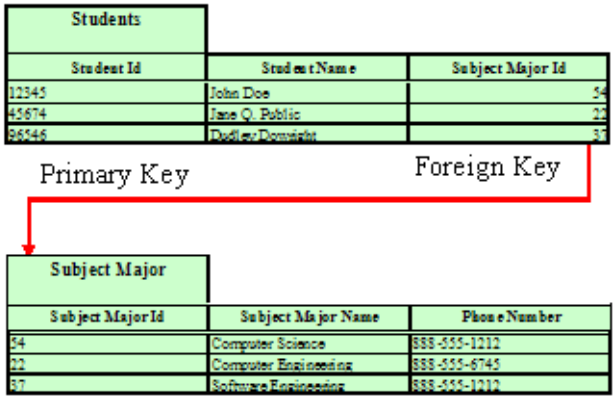
		defining the purpose, scope and information requirements of the architecture up-front, followed by collection of dat in accordance with a standard vocabulary. Data collected through the architectural process is delivered to the customer in either standard models or "Fit for Purpose" presentations. (Source DoD CIO promulgation memo, *The Department of Defense Architecture Framework (DoDAF) Version 2.0*, 28 May 2009; see the ASD(NII)/DoD CIO *Enterprise Architecture & Standards* site at http://cio-nii.defense.gov/policy/eas.shtml)
DoD Discovery Metadata Specification	DDMS	The DoD Discovery Metadata Specification (DDMS) defines discovery metadata elements for resources posted to community and organizational shared spaces. (Source: http://metadata.dod.mil/mdr/irs/DDMS/)
DoD Metadata Registry		As part of the overall **DoD Net-Centric Data Strategy**, the DoD CIO established the DoD Metadata Registry (http://metadata.dod.mil) and a related metadata registration process for the collection, storage and dissemination of structural metadata information resources (schemas, data elements, attributes, document type definitions, style-sheets, data structures, etc.). This Web-based repository is designed to also act as a clearinghouse through which industry and government coordination on metadata technology and related metadata issues can be advanced. As OASD's Executive Agent, **DISA** maintains and operates the ***DoD Metadata Registry and Clearinghouse*** under the direction and oversight of **OASD(NII)**. (Source: DoD Metadata Registry v6.0 Web site, https://metadata.dod.mil/mdr/about.htm)
DoD Net-Centric Data Strategy		This Strategy lays the foundation for realizing the benefits of net-centricity by identifying data goals and approaches for achieving those goals. To realize the vision for net-centric data, two primary objectives must be emphasized: (1) increasing the data that is available to communities or the Enterprise and (2) ensuring that data is usable by both anticipated and unanticipated users and applications. (Source: *Department of Defense Net-Centric Data Strategy*, DoD CIO, 9 May 2003, http://www.defenselink.mil/cio-nii/docs/Net-Centric-Data-Strategy-2003-05-092.pdf)
DoD PKI Class 3 Assurance Level		Applications handling unclassified medium value information in Moderately Protected Environments, unclassified high value information in Highly Protected Environments, and discretionary access control of classified information in Highly Protected Environments. This assurance level is appropriate for applications that require identification of an entity as a legal person, rather than merely as a member of an organization. ***Note:*** *This definition is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*
DoD PKI High Assurance		Applications that handle high value unclassified information (mission critical) in minimally protected environments require High Assurance certificates. Applications that are applicable for High Assurance certificates include the following: • All applications appropriate for DoD PKI Medium Assurance certificates

		• Digital signature services for unclassified Mission Assurance Category I (MAC I) or national security information in an unencrypted network • Protection (authentication and confidentiality) for information crossing classification boundaries when such a crossing is already permitted under a system security policy (e.g., sending unclassified information through a High Assurance Guard from **SIPRNet** to **NIPRNet**) (Source: adapted from ***X.509 Certificate Policy for the United States Department of Defense***, Version 9.0, 9 February 2005; http://iase.disa.mil/pki/dod-cp-v90-final-9-feb-05-signed.pdf; DoD PKI Certificate required)
Domain		A group of related items within a certain area of interest. In **DDS**, a domain is the basic construct used to bind individual **publications** and **subscriptions** together for communication. A distributed application can elect to use single or multiple domains for its **data-centric** communications. Domains isolate communication, promote scalability and segregate different classifications of data. (See **Global Data Space**)
Domain Analysis		The process of identifying the types of information that the data model uses. A good data model captures descriptive information about each of the types.
Domain Name System	DNS	The Domain Name System stores information about hostnames and domain names in a type of distributed database on networks, such as the Internet. Of the many types of information that can be stored, most importantly it provides a physical location (IP address) for each domain name, and lists the mail exchange servers accepting email for each domain. The DNS provides a vital service on the Internet as it allows the transmission of technical information in a user-friendly way. While computers and network hardware work with IP addresses to perform tasks such as addressing and routing, humans generally find it easier to work with hostnames and domain names (such as ***www.example.com***) in URLs and email addresses. The DNS therefore mediates between the needs and preferences of humans and of software.
Dual Stacking		Incorporating both IPv4 and IPv6 support in routers and computers.
Dynamic Host Configuration Protocol	DHCP	A protocol for assigning dynamic **Internet Protocol** (IP) addresses to devices on a network; DHCP a device can have a different IP address every time it connects to the network. (Source: http://www.webopedia.com/TERM/D/DHCP.html)
Dynamic HTML	DHTML	Designates a technique of creating interactive web sites by using a combination of the static markup language HTML, a client-side scripting language such as JavaScript, and the style definition language Cascading Style Sheets. (Source: http://en.wikipedia.org/wiki/Dynamic_web_page)
Dynamic Web Page		See **DHTML**.

Electronic Business Using eXtensible Markup Language	ebXML	ebXML is a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. Using ebXML, companies now have a standard method to exchange business messages, conduct trading relationships, communicate data in common terms and define and register business processes. (Source: http://www.ebxml.org/geninfo.htm)
Electronic Data Interchange	EDI	Standard formats for exchanging business data and documents.
Electronic Data Interchange Personnel Identifier	EDI-PI	A unique number assigned to each recipient of a **Common Access Card** (CAC), which is issued by the United States **Department of Defense** through the Defense Enrollment Eligibility Reporting System (DEERS). (Source: http://en.wikipedia.org/wiki/Electronic_Data_Interchange_Personal_Identifier)
Encryption		Encryption is the process of obscuring information to make it unreadable without special knowledge. While encryption has been used to protect communications for centuries, only organizations and individuals with an extraordinary need for secrecy have made use of it. In the mid-1970s, strong encryption emerged from the sole preserve of secretive government agencies into the public domain, and is now employed in protecting widely-used systems, such as Internet e-commerce, mobile telephone networks and bank automatic teller machines. (Source: http://en.wikipedia.org/wiki/Encryption)
Endpoint		The URL or location of the Web service on the internet.
End User		A human user of information. This is distinct from those who develop or support the automated systems that provide the information. -OR- A person who uses a device-specific user agent to access a Web site. (Source: http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf)
Enterprise		An organization considered as an entity or system that includes interdependent resources (e.g., people, organizations, and technology) that must coordinate functions and share information in support of a common mission or a set of related missions. In the computer industry, the term is often used to describe any large organization that utilizes computers. An intranet, for example, is a good example of an enterprise computing system. (Source: http://www.webopedia.com/TERM/e/enterprise.html)
Enterprise Application Integration	EAI	Software to effect interface between enterprise software systems. Provides interface at the application layer.
Enterprise Java Bean	EJB	A server-side component architecture for the development and deployment of object-oriented, distributed, enterprise-level applications. Applications written using the Enterprise JavaBeans architecture are scalable, transactional, and secure. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)

Enterprise Service		A service that provides capabilities to the enterprise. See also **Core Enterprise Service** and **Community of Interest Service**.
Enterprise Service Bus	ESB	An architectural style that provides distributed invocation, mediation, and end-to-end management and security of services and service interactions to support the larger architectural style known as Service Oriented Architecture (SOA) ***Note***: See the Enterprise Service Bus [P1389] in Part 5 for additional information.
Environment Variable		Environment variables are a set of dynamic values that can affect the way running processes will behave. (Source: http://en.wikipedia.org/wiki/Environment_variable)
eXtensible Access Control Markup Language	XACML	XACML is used to represent and evaluate access control policies. XACML is designed to standardize the use of declarative policy to control access to resources. Used with **SAML**.
eXtensible Markup Language	XML	A markup language defines tags (markup) to identify the content, data, and text in XML documents. It differs from **HTML**, the markup language most often used to present information on the Internet. HTML has fixed tags that deal mainly with style or presentation. An XML document must undergo a transformation into a language with style tags under the control of a style sheet before it can be presented by a browser or other presentation mechanism. Two types of style sheets used with XML are CSS and XSL. Typically, XML is transformed into HTML for presentation. Although tags can be defined as needed in the generation of an XML document, you can use a document type definition (DTD) to define the elements allowed in a particular type of document. A document can be compared by using the rules in the DTD to determine its validity and to locate particular elements in the document. A Web services application's J2EE deployment descriptors are expressed in XML with schemas defining allowed elements. Programs for processing XML documents use SAX or DOM APIs. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
eXtensible Stylesheet Language	XSL	Extensible Stylesheet Language (XSL) is a family of recommendations for defining XML document transformation and presentation. It consists of three parts: • XSL Transformations (XSLT): a language for transforming XML • XML Path Language (XPath): an expression language used by XSLT to access or refer to parts of an XML document • XSL Formatting Objects (XSL-FO): an XML vocabulary for specifying formatting semantics (Source: http://www.w3.org/Style/XSL/)
Facade		Provides a unified interface to a set of interfaces in a subsystem. Facade defines a higher-level interface that makes the subsystem easier to use. This can simplify a number of complicated object interactions into a single interface.

Facade Design Pattern		An object that provides a simplified interface to a larger body of code, such as a class library. (Source: http://en.wikipedia.org/wiki/Facade_pattern)
Federal Information Processing Standard	FIPS	Under the Information Technology Management Reform Act (Public Law 104-106), the Secretary of Commerce approves standards and guidelines that are developed by the National Institute of Standards and Technology (NIST) for Federal computer systems. These standards and guidelines are issued by NIST as Federal Information Processing Standards (FIPS) for use government-wide. NIST develops FIPS when there are compelling Federal government requirements such as for security and interoperability and there are no acceptable industry standards or solutions. (Source: http://www.itl.nist.gov/fipspubs/geninfo.htm)
Federated Search		Implementation of a computer program that allows users to access multiple data sources with a single query string located within a single interface. (Source: http://en.wikipedia.org/wiki/Federated_search)
File Transfer Protocol	FTP	FTP transfers files to and from a remote network. The protocol includes the ftp command (local machine) and the in.ftpd daemon (remote machine). FTP enables a user to specify the name of the remote host and file transfer command options on the local host's command line. The in.ftpd daemon on the remote host then handles the requests from the local host. Unlike RCP, FTP works even when the remote computer does not run a UNIX-based operating system. A user must log in to the remote computer to make an FTB connection unless it has been set up to allow anonymous FTP. (Source: http://www.sun.com/products-n-solutions/hardware/docs/html/817-6210-10/glossary.html)
Firewall		A piece of hardware and/or software which functions in a networked environment to prevent some communications forbidden by the security policy, analogous to the function of firewalls in building construction.
Font Size		The font size refers to the size of the font from baseline to baseline, when set solid (in CSS terms, this is when the ***font-size*** and ***line-height*** properties have the same value). (Source: http://www.w3.org/TR/REC-CSS2/fonts.html)
FORCEnet	Fn	An operational construct and architectural framework that integrates the SEAPOWER21 concepts of Sea Strike, Sea Shield, and Sea Basing by connecting warriors; sensors, networks; command and control; platforms and weapons; providing accelerated speed and accuracy of decision; and integrating knowledge to dominate the battlespace. FORCEnet provides the following capabilities: expeditionary, multi-tiered, sensor and weapon grids; distributed, collaborative, command and control; dynamic, multi-path survivable networks; adaptive/automated decision aids; and human-centric integration.
Foreign Key	FK	An attribute in a relation of a database that serves as the primary key of another relation in the same database.

		 11156
GIG Enterprise Service	GES	A service that provides capabilities for use in the DoD enterprise. GIG Enterprise Services are the combination of Core Enterprise Services and Community of Interest Services. Also referred to as Global Enterprise Services.
Global Command and Control System	GCCS	GCCS-J is the DOD joint C2 system of record for achieving full spectrum dominance. It enhances information superiority and supports the operational concepts of full-dimensional protection and precision engagement. GCCS-J is the principal foundation for dominant battlespace awareness, providing an integrated, near real-time picture of the battlespace necessary to conduct joint and multinational operations. It fuses select C2 capabilities into a comprehensive, interoperable system by exchanging imagery, intelligence, status of forces, and planning information. GCCS-J offers vital connectivity to the systems the joint warfighter uses to plan, execute, and manage military operations. GCCS-J is a Command, Control, Communications, Computer, and Intelligence (C4I) system, consisting of hardware, software, procedures, standards, and interfaces that provide a robust, seamless C2 capability. The system uses the Defense Information Systems Network (DISN) and must work over tactical communication systems to ensure connectivity with deployed forces in the tactical environment. (Source: http://www.disa.mil/gccs-j/)
Global Information Grid	GIG	Globally interconnected, end-to-end set of information capabilities, associated processes, and personnel for collecting, processing, storing, disseminating, and managing information on demand to warfighters, policy makers, and support personnel. The GIG includes all owned and leased communications and computing systems and services, software (including applications), data, security services, and other associated services necessary to achieve Information Superiority. It also includes National Security Systems (NSS) as defined in section 5142 of the Clinger-Cohen Act of 1996. The GIG supports all DoD, National Security, and related Intelligence Community (IC) missions and functions (strategic, operational, tactical, and business) in war and in peace. The GIG provides capabilities from all operating locations (bases, posts, camps, stations, facilities, mobile platforms, and deployed sites). The GIG provides interfaces to coalition, allied, and non-DoD users and systems.

Global Positioning System		A satellite constellation that provides highly accurate position, velocity, and time navigation information to users. (Source: JP 1-02,)
Graphical User Interface	GUI	A program that lets the user interact with a computer system in a highly visual manner, with a minimum of typing. Graphical user interfaces usually require a high-resolution display and a pointing device, such as a computer mouse. (Source: http://www.oreilly.com/catalog/debian/chapter/book/glossary.html)
Hard Code		To hard code or hard coding (also, hard-code/hard-coding, hardcode/hardcoding) refers to the software development practice of embedding output or configuration data directly into the source code of a program or other executable object, or fixed formatting of the data, instead of obtaining that data from external sources or generating data or formatting in the program itself with the given input. Considered an ***anti-pattern*** or ***Bad Thing***, hard coding requires the program's source code to be changed any time the input data or desired format changes, when it might be more convenient to the end user to change the detail by some means outside the program. (Source: http://en.wikipedia.org/wiki/Hard_code; 12 June 2007)
Hierarchical Database		A hierarchical database defines a set of parent-child relationships. Their use should be limited to integration of existing databases, such as IBM's Informational Management System (IMS). Hierarchical database systems require developers to predict all possible access patterns in advance and design the database accordingly. A database access pattern that is not included in the design becomes very difficult and inefficient.
High Assurance Internet Protocol Encryption	HAIPE	DoD version of Internet Protocol (IP) security (IPsec) protocol. (Source: http://en.wikipedia.org/wiki/HAIPE)
High Availability		Data tier availability can be affected by hardware failure, power outages, data errors, user errors, programmer errors, OS errors, and RDBMS errors. Various hardware and software methods help mitigate availability issues. The more reliable a system needs to be, the more it costs. Consequently, defining availability to meet requirements is essential to controlling costs.
Horizontal Fusion	HF	Horizontal Fusion (HF) is a direct response to Secretary of Defense Donald H. Rumsfeld's vision of Force Transformation. It demonstrates the ability to use lightweight automation to replace system mass with superior access to information based on a coherent architecture for an arbitrary future. Horizontal Fusion acts as a catalyst by implementing and demonstrating technologies and techniques that significantly advance the process of information-sharing in a an evolving net-centric environment. (Source: http://horizontalfusion.dtic.mil/vision/)
Hypertext Markup Language	HTML	A markup language for hypertext documents on the Internet. HTML supports embedding images, sounds, video streams, form fields, references to other objects with URLs, and basic text formatting. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)

Hypertext Transfer Protocol	HTTP	The Internet protocol used to retrieve hypertext objects from remote hosts. HTTP messages consist of requests from client to server and responses from server to client. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Hypertext Transmission Protocol Over SSL	HTTPS	HTTPS is the secure version of **HTTP**, the communication protocol of the World Wide Web. It was invented by Netscape Communications Corporation to provide authentication and encrypted communication and is used in electronic commerce. Instead of using plain text socket communication, HTTPS encrypts the session data using either a version of the **SSL** (Secure Socket Layer) protocol or the **TLS** (Transport Layer Security) protocol, thus ensuring reasonable protection from eavesdroppers, and man in the middle attacks. The default TCP/IP port of HTTPS is 443. (Source: http://en.wikipedia.org/wiki/HTTPS)
Identity		Identity refers to the nature or attributes of the track: ***Friend***, ***Assumed Friend***, ***Neutral***, ***Unknown***, ***Pending***, ***Suspect***, or ***Hostile***.
Identity Management		Provides the methodology and functions for maintaining information on people, consumers, and service providers. Supports the validation of identity authentication credentials.
Image Map		An image or graphic that has been coded to contain interactive areas. When it is clicked on, it launches another Web page or program. An image map usually has many different hyperlinked areas, known as links. For example, an image map of a country could be coded so that when a user clicks on a city or region, the browser is routed to a document or Web page about that place. (Source: http://www.netlingo.com/right.cfm?term=clickable%20graphic%20or%20imagemap)
Information		Data to which meaning is assigned, according to context and assumed conventions. Data that has been interpreted, translated, or transformed to reveal the underlying meaning.
Information Assurance	IA	Measures that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. This includes providing for restoration of information systems by incorporating protection, detection, and reaction capabilities. (Source: DoD Directive 8500.1, *Information Assurance (IA)*, http://www.dtic.mil/whs/directives/corres/pdf/850001p.pdf)
Information Support Plan	ISP	The identification and documentation of information needs, infrastructure support, IT and NSS interface requirements and dependencies focusing on net-centric, interoperability, supportability and sufficiency concerns. (Source: DoD Instruction 4630.8, 30 June 2004, [R1168] Enclosure 2, Definitions)

Information Technology	IT	Any equipment or interconnected system or subsystem of equipment, that is used in the automatic acquisition, storage, manipulation, management, movement, control, display, switching, interchange, transmission, or reception of data or information. Information technology includes computers, ancillary equipment, software, firmware, and similar procedures, services (including support services), and related resources. Information technology does not include any equipment that is acquired by a federal contractor incidental to a federal contract. (Source: CJCSI 6212.01E, [R1175] Glossary page GL-14)
Initial Capabilities Document	ICD	Documents the need for a materiel approach, or an approach that is a combination of materiel and non-materiel, to satisfy specific capability gap(s). It defines the capability gap(s) in terms of the functional area, the relevant range of military operations, desired effects, time and doctrine, organization, training, materiel, leadership and education, personnel, and facilities (DOTMLPF) and policy implications and constraints. The ICD summarizes the results of the DOTMLPF and policy analysis and the DOTMLPF approaches (materiel and non-materiel) that may deliver the required capability. The outcome of an ICD could be one or more joint DCRs or capability development documents. (Source: CJCSI 3170.01F, *Joint Capabilities Integration and Development System*, 1 May 2007, Glossary page GL-9)
Integrated Development Environment	IDE	
Integration		Integration is the action or process of combining elements so that they become a whole. Vertical integration acts within a system, whereas horizontal integration acts between or among systems. In the net-centric environment, integration creates links between computer systems, applications, services, or processes. The word is normally used in the context of computing, but can apply to business processes as much as to the underlying process automation. In the past, computer integration such as **enterprise application integration** (EAI) has typically been tightly coupled, or "hard wired," making it difficult to adapt to changing requirements. Thanks to the advent of Web services and the evolution of service-oriented architectures, more agile, loosely coupled forms of integration are starting to emerge.
Integrity		Quality of an Information System (IS) reflecting the logical correctness and reliability of the operating system; the logical completeness of the hardware and software implementing the protection mechanisms; and the consistency of the data structures and occurrence of the stored data. Note that, in a formal security mode, integrity is interpreted more narrowly to mean protection against unauthorized modification or destruction of information. R1339: Committee on National Security Systems (CNSS) Instruction 4009, **National Information Assurance (IA) Glossary** . [http://www.cnss.gov/Assets/pdf/cnssi_4009.pdf]
Intelligence Community	IC	A federation of executive branch agencies and organizations that conduct intelligence activities necessary for conduct of foreign relations and protection of national security. (Source: http://www.intelligence.gov/)

Interface		The functional and physical characteristics required to exist at a common boundary or connection between systems or items. (Source: *Defense Standardization Program (DSP) Policies and Procedures*, DoD 4120.24-M, March 2000) A Key Interface is a common boundary shared between system modules that provides access to critical data, information, materiel, or services; and/or is of high interest due to rapid technological change, a high rate of failure, or costliness of connected modules. (Source: *A Modular Open Systems Approach (MOSA) to Acquisition*, Version 2.0, September 2004; http://www.acq.osd.mil/osjtf/mosapart.html)
Interface Definition Language	IDL	A language used to define interfaces to remote **CORBA** objects. The interfaces are independent of operating systems and programming languages. (Source: http://java.sun.com/javaee/reference/glossary/index.jsp#120354)
International Telecommunication Union	ITU	United Nations agency for information and communication technologies. (Source: http://www.itu.int/net/about/index.aspx)
Internet		The Internet, or simply the Net, is the publicly available worldwide system of interconnected computer networks that transmit data by packet switching using a standardized Internet Protocol (IP) and many other protocols. It is made up of thousands of smaller commercial, academic, and government networks. It carries various information and services, such as electronic mail, online chat and the interlinked web pages and other documents of the World Wide Web. Because this is by far the largest, most extensive internet (with a lower case i) in the world, it is simply called the Internet (with a capital I). (Source: http://en.wikipedia.org/wiki/Internet)
Internet Engineering Task Force	IETF	The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. It is open to any interested individual. (Source: http://www.ietf.org/overview.html)
Internet Information Services	IIS	A set of Internet-based services for Windows machines. Originally supplied as part of the Option Pack for Windows NT, they were subsequently integrated with Windows 2000 and Windows Server 2003. The current (Windows 2003) version is IIS 6.0 and includes servers for FTP, SMTP, NNTP and HTTP/HTTPS. Earlier versions also included a Gopher server.
Internet Inter-ORB Protocol	IIOP	A protocol used for communication between CORBA object request brokers. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Internet Protocol	IP	Data packets routed across network, not switched via dedicated circuits.
Internet Protocol Version 4	IPv4	Version 4 of the Internet Protocol (IP). It was the first version of the Internet Protocol to be widely deployed, and forms the basis for most of the current Internet (as of 2004). It is described in IETF RFC 791, which

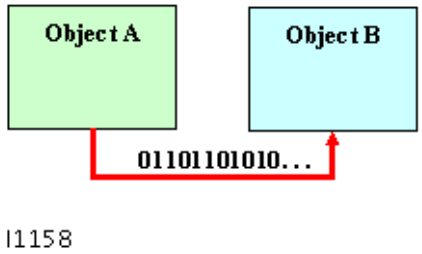
		was first published in September, 1981. IPv4 uses 32-bit addresses, limiting it to 4,294,967,296 unique addresses, many of which are reserved for special purposes such as local networks or **multicast** addresses. This reduces the number of addresses that can be allocated as public Internet addresses. As the number of addresses available is consumed, an IPv4 address shortage appears to be inevitable in the long run. This limitation has helped stimulate the push towards IPv6, which is currently in the early stages of deployment, and may eventually replace IPv4. (Source: http://en.wikipedia.org/wiki/IPv4)
Internet Protocol Version 6	IPv6	Version 6 of the Internet Protocol; it was initially called IP Next Generation (IPng) when it was picked as the winner in the IETF's IPng selection process. IPv6 is intended to replace the previous standard, IPv4, which only supports up to about 4 billion (4×10^9) addresses. IPv6 supports up to about 3.4×10^{38} (340 undecillion) addresses. This is the equivalent of 4.3×10^{20} (430 quintillion) addresses per square inch (6.7×10^{17} (670 quadrillion) addresses/mm^2)of the Earth's surface. It is expected that IPv4 will be supported until at least 2025, to allow time for bugs and system errors to be corrected. (Source: http://en.wikipedia.org/wiki/Ipv6)
Interoperability		The ability of systems, units, or forces to provide data, information, materiel, and services to and accept the same from other systems, units, or forces, and to use the data, information, materiel, and services so exchanged to enable them to operate effectively together. **IT** and **NSS** interoperability includes both the technical exchange of information and the end-to-end operational effectiveness of that exchanged information as required for mission accomplishment. Interoperability is more than just information exchange. It includes systems, processes, procedures, organizations, and missions over the life cycle and must be balanced with **IA**. (Source: CJCSI 6212.01E, ***Interoperability and Supportability of Information Technology and National Security Systems***, 15 December 2008)
Intranet		An intranet is a local area network (LAN) used internally in an organization to facilitate communication and access to information that is sometimes access-restricted. Sometimes the term refers only to the most visible service, the internal web site. The same concepts and technologies of the Internet such as clients and servers running on the Internet protocol suite are used to build an intranet. HTTP and other internet protocols are commonly used as well, especially FTP and email. There is often an attempt to use internet technologies to provide new interfaces with corporate "legacy" data and information systems. (Source: http://en.wikipedia.org/wiki/Intranet)
Intrusion Detection System	IDS	An IDS inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. (Source: http://www.webopedia.com/TERM/i/intrusion_detection_system.html)
ISO/IEC 11179		ISO-11179 (formally known as the ISO/IEC 11179 Metadata Registry (MDR) Standard) is the international standard for representing **metadata** for an organization in a **Metadata Registry**. (Source: http://en.wikipedia.org/wiki/ISO/IEC_11179)

J2EE Server		The runtime portion of a J2EE product. A J2EE server provides EJB or Web containers or both. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Java		Java is a reflective, object-oriented programming language developed initially by at Sun Microsystems. It was intended to replace C++, although the feature set better resembles that of Objective-C. Java should not be confused with JavaScript, which shares only the name and a similar C-like syntax. Sun Microsystems currently maintains and updates Java regularly. Specifications of the Java language, the Java Virtual Machine (JVM) and the Java API are community-maintained through the Sun-managed Java Community Process.
Java 2 Platform, Enterprise Edition	J2EE	The J2EE environment is the standard for developing component-based multi-tier enterprise applications. The J2EE platform consists of a set of services, application programming interfaces (APIs), and protocols that provide the functionality for developing multitiered, Web-based applications. Features include Web services support and development tools. Sun Microsystems has simplified the name of the Java platform for the enterprise; the "2" is dropped from the name, as well as the dot number so the next version of the Java platform for the enterprise is **Java Platform, Enterprise Edition** 5 or Java EE 5.(Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Java Archive	JAR	A platform-independent file format that enables you to bundle multiple files into a single archive file. JAR files are packaged with the ZIP file format, so you can use them for ZIP-like tasks such as lossless data compression, archiving, decompression, and archive unpacking. Typically JAR files contain the class files and auxiliary resources associated with applets and applications. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Java Class Files		Class files contain bytecodes for the Java Virtual Machine. They are normally produced by a compiler for the Java programming language. A Java interpreter can then read these files and execute the code contained within.
Java Database Connection	JDBC	An API that supports database and data-source access from Java applications.
Java Development Kit	JDK	
Javadoc		Javadoc is a computer software tool from Sun Microsystems for generating API documentation into HTML format from Java source code. Javadoc is the industry standard for documenting Java classes. Most **Integrated Development Environments** (**IDEs**) will automatically generate Javadoc HTML. (Source: http://en.wikipedia.org/wiki/Javadoc)
Java Message Service	JMS	An API for invoking operations on enterprise messaging systems. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)

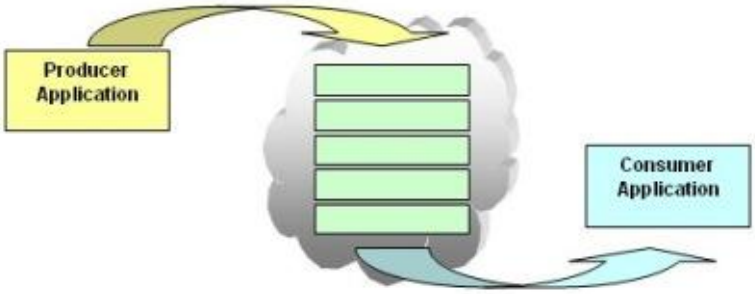
Java Naming and Directory Interface	JNDI	An API that provides naming and directory functionality. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Java Platform, Enterprise Edition	Java EE	Java Platform, Enterprise Edition (Java EE) is the industry standard for developing portable, robust, scalable and secure server-side Java applications. Building on the solid foundation of the Java Platform, Standard Edition (Java SE), Java EE provides Web services, component model, management, and communications APIs that make it the industry standard for implementing enterprise-class service-oriented architecture (SOA) and next-generation Web applications. Sun Microsystems has simplified the name of the Java platform for the enterprise. Formerly, the platform was known as Java 2 Platform, Enterprise Edition (**J2EE**), and specific versions had "dot numbers" such as J2EE 1.4. The "2" is dropped from the name, as well as the dot number so the next version of the Java platform for the enterprise is Java Platform, Enterprise Edition 5 or Java EE 5. (Source: http://java.sun.com/javaee/)
JavaScript		The Netscape-developed object scripting language used in millions of web pages and server applications worldwide. Contrary to popular misconception, JavaScript is not "Interpretive Java." Rather, it is a dynamic scripting language that supports prototype-based object construction.
JavaServer Pages	JSP	An extensible Web technology that uses static data, JSP elements, and server-side Java objects to generate dynamic content for a client. Typically the static data is HTML or XML elements, and in many cases the client is a Web browser. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Joint Capabilities Integration and Development System	JCIDS	Establishes procedures to support the Chairman of the Joint Chiefs of Staff and the Joint Requirements Oversight Council (JROC) in identifying, assessing and prioritizing joint military capability. (Source: CJCSI 3170.01F, 1 May 2007, *Joint Capabilities Integration and Development System*)
Joint Interoperability Test Command	JITC	JITC provides a full-range of agile and cost-effective test, evaluation, and certification services to support rapid acquisition and fielding of global net-centric warfighting capabilities. (Source: http://jitc.fhu.disa.mil/mission.html)
Joint Tactical Radio System	JTRS	JTRS is a family of interoperable, affordable software defined radios which provide secure, wireless networking communications capabilities for Joint forces. (Source: JTRS JPEO, http://jpeojtrs.mil/)
Joint Worldwide Intelligence Communications System	JWICS	The sensitive compartmented information portion of the **Defense Information Systems Network**. It incorporates advanced networking technologies that permit point-to-point or multipoint information exchange involving voice, text, graphics, data, and video teleconferencing. (Source:)

JScript		Microsoft's extended implementation of ECMAScript (ECMA262), an international standard based on Netscape's JavaScript and Microsoft's JScript languages. JScript is implemented as a Windows Script engine. This means that you can plug it in to any application that supports Windows Script, such as Internet Explorer, Active Server Pages, and Windows Script Host. It also means that any application supporting Windows Script can use multiple languages: JScript, VBScript, Perl, and others.
Just-In-Time Compilation	JIT	This is the primary method by which **.NET** executes **MSIL**. As the MSIL is executed, the code is compiled and optimized for the executing environment. JIT compilation provides environment optimization, runtime type safety, and assembly verification. To accomplish this, the JIT compiler examines the assembly metadata for any illegal accesses and handles violations appropriately.
Key Interface Profile	KIP	An operational functionality, systems functionality and technical specifications description of the Key Interface. The profile consists of refined Operational and Systems Views, interface control specifications, Technical View with SV-TV Bridge, and referenced procedures for KIP compliance. The key interface profile is the technical specification that governs access to the **GIG**. (Source: CJCSI 6212.01D, 8 March 2006, Glossary page GL-14) ***Note:*** *CJCSI 6212.01E[R1175], 15 December 2008, deletes the "Key Interface Profile" element of the NR-KPP and replaces it with the "Technical Standards/Interfaces" element. This revision further indicates that Global Information Grid (GIG) Enterprise Service Profiles (GESPs) are evolving to provide a net-centric oriented approach for managing interoperabilty across the GIG based on the definition and configuration control of key interfaces and enterprise services.*
Key Performance Parameters	KPP	Those attributes or characteristics of a system that are considered critical or essential to the development of an effective military capability and those attributes that make a significant contribution to the key characteristics as defined in the Joint Operations Concepts. KPPs are validated by the Joint Requirements Oversight Council (JROC) for JROC Interest documents, and by the DOD component for Joint Integration or Independent documents. Capability development and capability production document KPPs are included verbatim in the acquisition program baseline. (Source: CJCSI 3170.01F[R1173], *Joint Capabilities and Development System*, 1 May 2007, Glossary page GL-14)
Key Recovery Manager	KRM	A service of the DOD PKI where copies of key pairs used for encryption are stored and can be recovered for law enforcement purposes. ***Note:*** *This definition is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*
Keystore		A file containing the keys and certificates used for authentication. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)

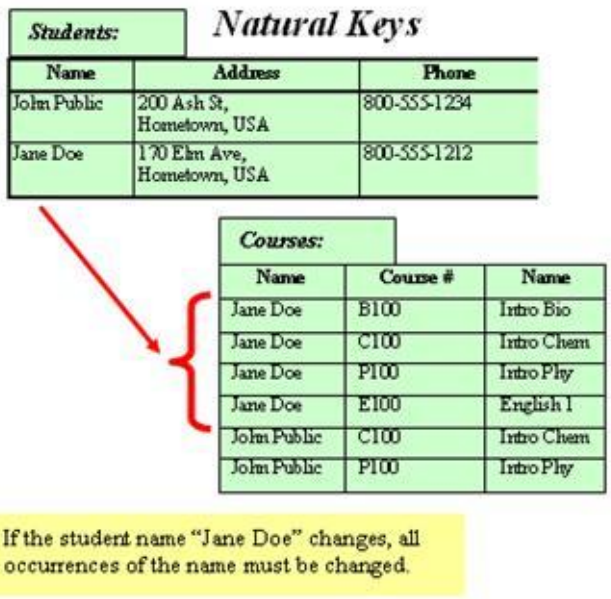
Knowledge		(Unlike information or data) Requires the presence of context, semantics, and purpose.
Least-Common-Denominator Data Access Mechanism		When one application is able to obtain data provided by another by removing arbitrary implementation barriers to data exchange.
Legacy System		An existing computer system or application program which continues to be used because the user (typically an organization) does not want to replace or redesign it. (Source: http://en.wikipedia.org/wiki/Legacy_system)
Light Directory Access Protocol	LDAP	A set of protocols for accessing information directories. LDAP is a simpler version of the X.500 standard. Unlike X.500, LD Web Services for Interactive Applications AP supports TCP/IP, which is necessary for Internet access. Because it's a simpler version of X.500, LDAP is sometimes called X.500-lite. LDAP is a protocol for accessing on-line directory services. (Source: http://en.wikipedia.org/wiki/LDAP)
Link-16	TADIL-J	Tactical Data Information Link (TADIL) primarily designed for use by Command and Control (C2) and Air-to-Air assets; uses the Joint Tactical Data Link (TADIL-J) message format. (Source: http://aatc.aztucs.ang.af.mil/aatcinfo.htm)
Linked Style Sheets		Style sheets that are placed in a separate text files and saved in the root with a css file extension. A link to the file is made in the head section of the document. `<head><Break/> <link<Break/> rel="stylesheet"<Break/> href="mystyle.css"<Break/> type="text/css"><Break/></head><Break/>`
Local Area Network	LAN	A group of interconnected computer and support devices. (Source: http://www.sun.com/products-n-solutions/hardware/docs/html/817-6210-10/glossary.html)
Look and Feel		Look and feel refers to design aspects of a graphical user interface in terms of colors, shapes, layout, typefaces, etc. (the "look"); and, the behavior of dynamic elements such as buttons, boxes, and menus (the "feel"). It is used in reference to both software and **Web sites**. (Source: http://en.wikipedia.org/wiki/Look_and_feel)
Loosely Coupled		A computing model where application elements require a simple level of coordination and allow for flexible reconfiguration. Interconnection is often asynchronous and message-based.
Machine-to-Machine Messaging		Provides reliable machine-to-machine message exchange across the **enterprise**.

Marshalling		The process of transferring data using **serialization** and **deserialization** is called marshalling. 
Mediation		A set of negotiated agreements for interacting between components that enable those components to work together to perform a task. These agreements are defined through standard interfaces and data interchange specifications. Mediation services provide multiple methods for integrating data sources and services: **Transformation**: When a client requests data from a service in a particular format, a transformer retrieves and reformats the data before returning it to the client **Aggregation**: A mediator service may collect data derived from multiple sources, thus making many services appear to be one **Adaptation**: When a client cannot communicate directly with a service, an adapter provides service mediation (can be transport protocol as well as data format) when services need to communicate point-to-point **Orchestration**: Co-ordination of events in a process; orchestration directs and manages the on-demand assembly of multiple component services to create a composite application or business process **Choreography**: When a client request spawns a chain of events or service requests that do not rely on a central coordinator, a Choreographed Web Service knows when to execute other services and with which other services to interact; WS-CDL is an example of a business process management workflow language that implements choreography
Message		A self-contained unit of information exchanged between a producer and one or more consumers. Software commonly uses messages to communicate synchronously or asynchronously between service producers and consumers. Some examples of software messaging are **SOAP** messages, e-mail messages, **Data Distribution Service** (**DDS**) messages, and **Java Message Service** (**JMS**) messages.

Message-Oriented Middleware	MOM	Message-oriented middleware acts as an arbitrator between incoming and outgoing messages to insulate producers and consumers from other producers and consumers.
Metadata		Data about the data, that is, the description of the data resources, its characteristics, location, usage, and so on. Metadata is used to identify, describe, and define user data.
Metadata Registry		A Metadata Registry is a central place where metadata definitions are stored and maintained. A metadata registry typically has the following characteristics: • It is a protected area where only approved individuals may make changes • It stores data elements that include both semantics and representations • The semantic areas of a metadata registry contain the meaning of a Data Element with precise definitions • The representational areas define how the data is represented in a specific format such as within a database or a structure file format such as XML Metadata Registries often are stored in an international format called **ISO-11179**.
Microsoft Intermediate Language	MSIL	An intermediate instruction set into which all **.NET** languages compile. You can execute MSIL code on any environment that supports the .NET framework. MSIL-compiled code is verified for safety during runtime, providing better security and reliability than natively compiled binaries. During compilation, .NET code is translated into Microsoft Intermediate Language (MSIL) rather than machine-specific binary code. MSIL is a machine- and platform-independent instruction set that can be executed in any environment within the .NET framework. .NET uses **just-in-time (JIT) compilation** as its primary means of executing MSIL. You can generate native binary images using Microsoft's Native Image Generator (**NGEN**).
Microsoft Message Queue	MSMQ	Messaging in .NET uses Microsoft Message Queue (MSMQ). MSMQ is responsible for reliably delivering messages between applications inside and outside the enterprise. MSMQ ensures reliable delivery by placing messages that fail to reach their intended destination in a queue and then resending them once the destination is reachable.

		 I1067 MSMQ also supports transactions. It permits multiple operations on multiple queues, with all of the operations wrapped in a single transaction, thus ensuring that either all or none of the operations will take effect. Microsoft Distributed Transaction Coordinator (MSDTC) supports transactional access to MSMQ and other resources.
Model-Driven Architecture	MDA	Model-driven architecture is a trademarked term denoting a specific approach to the development of software using models as the basis. The MDA specifies system functionality separately from the implementation of that functionality on a specific technology platform. To accomplish this goal, the MDA defines an architecture that provides a set of guidelines for structuring specifications expressed as models. The MDA model architecture relates multiple standards, including Unified Modeling Language (UML), the Meta Object Facility (MOF), the XML Metadata interchange (XMI), and the Common Warehouse Metamodel (CWM). Note that the term "architecture" in MM does not refer to the architecture of the system being modeled, but rather to the architecture of the various standards and model forms that serve as the technology basis for MDA .
Modular Design		Characterized by (1) Functional partitioning into discrete scalable, reusable modules consisting of isolated, self-contained functional elements; (2) Rigorous use of well-defined modular interfaces, including object-oriented descriptions of module functionality; (3) Ease of change to achieve technology transparency and, to the extent possible, make use of industry standards for key interfaces.
Module		(1) A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to, or output from, an assembler, compiler, linkage editor, or executive routine. (2) A logically separable part of a program. Note: The terms ***module***, ***component***, and ***unit*** are often used interchangeably or defined to be sub-elements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized. See also **component**. (Source: IEEE Std 610.12-1990)
Multicast		The delivery of information to a group of destinations simultaneously using the most efficient strategy to deliver the messages over each link of the network only once and only create copies when the links to the destinations split. (Source: http://en.wikipedia.org/wiki/Multicast)
MX Record		An MX record or Mail exchanger record is a type of resource record in the **Domain Name System** (DNS) specifying how **Internet** e-mail should

		be routed. MX records point to the servers that should receive an e-mail, and their priority relative to each other. (Source: http://en.wikipedia.org/wiki/MX_Record)
Namespace		A namespace is an abstract container which contains a logical grouping of unique identifiers (i.e., names). An identifier defined in a namespace is associated with that namespace. It is possible to define the same identifier independently in multiple namespaces. That is, the meaning associated with an identifier defined in one namespace may or may not have the same meaning as the same identifier defined in another namespace. Languages that support namespaces specify the rules that determine to which namespace an identifier (i.e., not its definition) belongs. (Adapted from: http://en.wikipedia.org/wiki/Namespace_%28computer_science%29; accessed 2/6/2008) XML namespaces provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by URI references. (Source http://www.w3.org/TR/REC-xml-names/)
National Institute of Standards and Technology	NIST	Non-regulatory federal agency within the U.S. Commerce Department's Technology Administration with a mission to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve our quality of life. (Source: http://www.nist.gov/public_affairs/general2.htm)
National Security Agency	NSA	America's cryptologic organization; it coordinates, directs, and performs highly specialized activities to protect U.S. government information systems and produce foreign signals intelligence information. (Source: http://www.nsa.gov/about/index.cfm)
National Security Systems	NSS	Telecommunications and information systems, operated by the Department of Defense, the functions, operation, or use of which involves: (1) intelligence activities; (2) cryptologic activities related to national security; (3) the command and control of military forces; (4) equipment that is an integral part of a weapon or weapons systems; or (5) is critical to the direct fulfillment of military or intelligence missions. Subsection (5) in the preceding sentence does not include procurement of automatic data processing equipment or services to be used for routine administrative and business applications (including payroll, finance, logistics, and personnel management applications). (Source: CJCSI 3170.01F, 1 May 2007, page GL-16)
Native Image Generator	NGEN	NGEN compilation enables you to production of a native binary image of **MSIL** code for the current environment. This improves the performance of the **.NET** application by eliminating the **JIT** overhead associated with the execution. Running NGEN against an assembly, the resulting native image is placed in the Global Assembly Cache for use by all other .NET assemblies. NGEN is a good tool for improving performance of .NET applications as long as the executing environment remains static. If executing an NGEN-generated image in an incompatible environment, .NET automatically reverts to using JIT. To mitigate this, run NGEN during deployment against the installed assemblies.

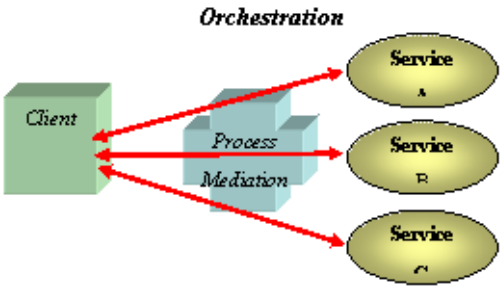
Native XML Database		Defines a logical model for an XML document (as opposed to the data in that document) and stores and retrieves documents according to that model. These databases are accessed via programming interfaces such as SAX, DOM, or JDOM. There is a trend away from pure XML storage because all the leading relational database vendors are introducing advanced XML capabilities.
Natural Key		A Natural Key is a primary keys that is made up completely or in part from naturally occurring data in the tables.  I1163 See **Surrogate Key** and **Primary Key**.
Net-Centric		Information-based operations that use service-oriented information processing, networks, and data from the following perspectives: user functionality (capability to adaptively perform assigned operational roles with increasing use of system-provided intelligence/cognitive processes), interoperability (shared information and loosely coupled services), and enterprise management (net operations). (Source: DoD Instruction 4630.8, Procedures for Interoperability and Supportability of Information Technology (IT) and National Security Systems (NSS), June 30, 2004 [R1168])
Net-Centric Enterprise Services	NCES	The NCES program provides enterprise-level Information Technology (IT) services and infrastructure components, also called Core Enterprise Services, for the Department of Defense (DoD) Global Information Grid (GIG).
Net-Centric Operations and Warfare Reference Model	NCOW RM	The NCOW RM described the activities required to establish, use, operate, and manage the net-centric enterprise information environment to include the generic user interface, the intelligent-assistant capabilities, the net-centric service capabilities (core services, **Community of Interest (COI) services**, and environment control services), and the enterprise management components. It also described a selected set of key standards that would be needed as the NCOW capabilities of

		the **Global Information Grid** (GIG) were realized. The NCOW RM represented the objective end-state for the GIG: a service-oriented, inter-networked, information infrastructure in which users request and receive services that enable operational capabilities across the range of military operations; **DoD** business operations; and Department-wide enterprise management operations. The NCOW RM was a key compliance mechanism for evaluating DoD information technology capabilities and the **Net-Ready Key Performance Parameter** in accordance with Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01D, *Interoperability and Supportability of Information Technology and National Security Systems*, 8 March 2006. The 15 December 2008 revision to this instruction, CJCSI 6212.01E, removed the NCOW RM element of the Net-Ready Key Performance Parameter (NR-KPP), integrating the components of the former NCOW RM into other elements of the NR-KPP. (Source: CJCSI 6212.01E [R1175])
Net-Ready Key Performance Parameter	NR-KPP	The NR-KPP is a key parameter stating a system's information needs, information timeliness, information assurance (IA), and net-ready attributes required for both the technical exchange of information needs, information timeliness, IA, and net-ready attributes required for both the technical exchange of information and the operational effectiveness of that exchange. The NR-KPP consists of information required to evaluate the timely, accurate, and complete exchange and use of information to satisfy information needs for a given capability. ***Note:*** *The 15 December 2008 revision of the Chairman Joint Chief of Staff Instruction for Interoperability and Supportability of Information Technology and National Security Systems (CJCSI 6212.01E) removed the* ***NCOW RM*** *element of the NR-KPP, integrating its components into the other elements of the NR-KPP.* The NR-KPP is composed of the following five elements: • Compliant solution architecture • Compliance with DOD Net-Centric Data [R1172] and Services [R1313] strategies, including data and services exposure criteria • Compliance with applicable GIG Technical Direction to include **DISR**-mandated IT Standards reflected in the TV-1 and implementation guidance of GIG Enterprise Service Profiles (GESPs) necessary to meet all operational requirements specified in the DOD Information Enterprise Architecture and solution architecture system/service views • Verification of compliance with DOD IA requirements • Compliance with supportability elements to include, spectrum analysis, Selective Availability Anti-Spoofing Module (SAASM), and the Joint Tactical Radio System (JTRS) (Source: CJCSI 6212.01E [R1175])
Network Centric Warfare	NCW	NCW is an information superiority-enabled concept of operations that generates increased combat power by networking sensors, decision makers, and shooters to achieve shared awareness, increased speed of command, higher tempo of operations, greater lethality, increased survivability, and a degree of selfsynchronization. In essence, NCW translates information superiority into combat power by effectively linking

		knowledgeable entities in the battlespace. (Source: *Network Centric Warfare: Developing and Leveraging Information Superiority*. David S. Alberts, John J. Garstka and Frederick P. Stien. DoD Command and Control Research Program Publication Series, available at http://www.dodccrp.org/files/Alberts_NCW.pdf)
Network Intrusion Detection	NID	Attempt to detect malicious activity such as denial of service attacks, port-scans or even attempts to crack into computers by monitoring network traffic. (Source: http://en.wikipedia.org/wiki/Network_intrusion-detection_system)
Network Operations	NetOps	An organizational, procedural, and technological construct for ensuring information and decision superiority at the strategic, operational, and tactical levels of warfare as well as within DoD business operations. NetOps is an operational approach, which addresses the interdependency and integration of IA/CND, S&NM, and CS capabilities. NetOps consists of the organizations, tactics, techniques, procedures, functionalities, and technologies required to plan, administer, and monitor use of the GIG infrastructure and the end-to-end information flows of the GIG; and to respond to threats, outages, and other operational impact. NetOps ensures mission requirements are properly considered in GIG operational decision-making. NetOps enables the GIG to provide its users with information they need, when and where they need it, with appropriate protection. NetOps is essential for successful execution of net-centric warfare and other net-centric operations in support of national security objectives.
Network Time Protocol	NTP	Protocol for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks. NTP uses **User Datagram Protocol** (UDP) port 123 as its transport layer. It is designed particularly to resist the effects of variable latency. (Source: http://en.wikipedia.org/wiki/Network_Time_Protocol)
Node		In general network usage, a node is a processing location such as a computer or some other device. Every node has a unique network address, sometimes called a Data Link Control (DLC) address or Media Access Control (MAC) address. (Source: http://www.webopedia.com/TERM/n/node.html) A NESI Node is a collection of integrated components (i.e., systems, applications, services and other Nodes) that are bound together spatially and/or temporally to meet the needs of a particular mission. It is conceptual in nature and can not be defined in terms of a concrete set of components or size. The membership of a component within a particular Node is not exclusive and a Component can be part of multiple Nodes.
Node Information Services	NIS	
Nonce		A unique random string.
Normalization		Normalization avoids duplication of data, insert anomalies, delete anomalies, and update anomalies. A relation is in first normal form (1NF) if and only if all underlying simple domains contain atomic values only. A relation is in second normal form (2NF) if and only if it is in 1NF and

		every non-key attribute is fully dependent on the primary key. A relation is in third normal form (3NF) if and only if it is in 2NF and every non-key attribute is non-transitively dependent on the primary key. Data models should follow the three forms unless there is overriding justification not to. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Object Management Group	OMG	OMGTM is an international, open membership, not-for-profit computer industry consortium. OMG Task Forces develop enterprise integration standards for a wide range of technologies, and an even wider range of industries. OMG's modeling standards enable powerful visual design, execution and maintenance of software and other processes. OMG's middleware standards and profiles are based on the **Common Object Request Broker Architecture** (CORBA) and support a wide variety of industries. (Source: http://www.omg.org/)
Object-Oriented Analysis	OOA	OOA (Object Oriented Analysis) constitutes the development of software engineering requirements and specifications for a system. These are expressed as an object model (object oriented design) which is composed of a population of interacting objects.
Object-Oriented Databases	OODBMS	Object-oriented databases are based on the object model, and use the same conceptual models as **object-oriented analysis** and **design**.
Object-Oriented Design		Any design that incorporates objects, classes, and inheritance. Contrast with object-based design and class-based design.
Object-Oriented Programming Language		A programming language that enables programmers to define and use objects, classes, and inheritance; for example, C++, Ada 95.
Object Request Broker	ORB	A library that enables **CORBA** objects to locate and communicate with one another. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Online Certificate Status Protocol	OCSP	Online Certificate Status Protocol is a method for determining the revocation status of an X.509 digital certificate using means other than **CRLs**. It is described in RFC 2560 and is on the Internet standards track. OCSP messages are encoded in ASN.1 and usually communicated over **HTTP**. OCSP's request/response nature leads to OCSP servers being termed as OCSP responders.
Online Status Check	OSC	OSC is service that may be provided by the **Certificate Authority** (CA). A relying party sends a request to the OSC service with a certificate, the OSC service responds with a digitally signed response that includes the date and time, certificate identification, and the status of the certificate about whose validity the relying party inquired. The possible responses include "unknown" which may be the response to a query regarding an expired certificate.

		Note: *This definition is derived from the DoD Class 3 PKI Public Key-Enabled Application Requirements Document, Version 1.0, 13 July 2000.*
Online Status Check Responder	OSCR	OSCR is the server that responds to a relying party's OSC request.
Ontology		An explicit specification of how to represent the objects and concepts that exist in some area of interest and of the relationships that pertain among them. (Source: DoD 8320.02-G, 12 April 2006, Guidance for Implementing Net-Centric Data Sharing)
Open Database Connectivity	ODBC	In computing, Open Database Connectivity (ODBC) provides a standard software API method for using database management systems (DBMS). The designers of ODBC aimed to make it independent of programming languages, database systems, and operating systems. (Source: http://en.wikipedia.org/wiki/Odbc; 30 March 2007)
Open Standard		Open standards are publicly available specifications for achieving a specific task. By allowing anyone to obtain and implement the standard, they can increase compatibility between various hardware and software components, since anyone with the necessary technical know-how and resources can build products that work together with those of the other vendors that base their designs on the standard (although patent holders may impose "reasonable and non-discriminatory" royalty fees and other licensing terms on implementers of the standard). Source: http://en.wikipedia.org/wiki/Open_standard) ***Note:*** *NESI restricts the use of the term "standard" to technologies approved by formalized committees that are open to participation by all interested parties and operate on a consensus basis.*
Open System		An open system employs modular design, uses widely supported and consensus based standards for its key interfaces, and has been ubjected to successfully validation and verification test to nsure the openness of its key interfaces. (Source: Open Systems Joint Task Force Program Manager's Guide, ***A Modular Open Systems Approach to Acquisittion***, Version 2.0, September 2004; Appendix A - Definitions)
Operational View	OV	The OV is a description of the tasks and activities, operational elements, and information exchanges required to accomplish DoD missions. DoD missions include both warfighting missions and business processes. The OV contains graphical and textual products that comprise an identification of the operational nodes and elements, assigned tasks and activities, and information flows required between nodes. It defines the types of information exchanged, the frequency of exchange, which tasks and activities are supported by the information exchanges, and the nature of information exchanges. (Source: *DoDAF* v1.5 Volume I: Definitions and Guidelines, 23 April 2007)
Orchestration		Co-ordination of events in a process; orchestration directs and manages the on-demand assembly of multiple component services to

		create a composite application or business process. (Source: http://looselycoupled.com/glossary/orchestration)  I1164 **Note:** *See **Mediation**.*
Organization for the Advancement of Structured Information Standards	OASIS	A not-for-profit, international consortium that drives the development, convergence, and adoption of e-business standards. (Source: http://www.oasis-open.org/who/)
OS File Systems		A file system that stores and retrieves data, acting as a data tier. Advocates cite performance and simplicity, but the loss of DBMS-inherent capabilities such as ad-hoc queries and the ability to upgrade to faster machines is a deterrent. File-system-based data tiers often result in proprietary solutions that are hard to maintain and port.
OWL		See **Web Ontology Language** entry.
Parser		A module that reads in XML data from an input source and breaks it into chunks so that your program knows when it is working with a tag, an attribute, or element data. A non-validating parser ensures that the XML data is well formed but does not verify that it is valid. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Personalization		The ability for portal members to subscribe to specific types of content and services. Users can customize the look and feel of their environment.
Personally Identifiable Information	PII	Personally Identifiable Information is any information about an individual maintained by an agency, including, but not limited to, education, financial transactions, medical history, and criminal or employment history and information which can be used to distinguish or trace an individual's identity. such as their name, social security number, data and place of birth, mother's maiden name, biometric records, etc., including any other personal information which is linked or linkable to an individual. Source: Department of Defense Guidance on Protecting Personally Identifiable Information (PII)

		R1332: DoD Memorandum , **Department of Defense Guidance on Protecting Personally Identifiable Information (PII)** . [http://iase.disa.mil/policy-guidance/pii-signed-memo-08182006.pdf]
Personal Web Server	PWS	A **Web server** program for personal computer users who want to share **Web pages** and other files from their hard drive. PWS is a scaled-down version of Microsoft's more robust Web server, Internet Information Server (**IIS**). PWS can be used with a full-time **Internet** connection to serve Web pages for a **Web site** with limited traffic. It can also be used for testing a Web site offline or from a "staging" site before putting it on a main Web site that is exposed to more traffic.
Physical Model		Translates the conceptual model to a particular RDBMS implementation.
Plain Text	PT	Textual data in **ASCII** format. Plain text is the most portable format because it is supported by nearly every application on every machine. It is quite limited, however, because it cannot contain any formatting commands. In cryptography, plain text refers to any message that is not encrypted. (Source: http://www.webopedia.com/TERM/p/plain_text.html)
Portability		The ease with which a system or component can be transferred from hardware or software environment to another. (Source: IEEE Std 610.12-1990) The level of software portability of any specific product depends on two factors: the design of the product itself, and the characteristics of the source and target execution environments. Software products are rarely if ever 100% portable. Generally, the level of portability depends on the target platform. Software that is highly portable to one class of platform might be not portable to other classes.
Portable Object Adapter	POA	A CORBA standard for building server-side applications that are portable across heterogeneous ORBs. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Portable Operating System Interface for Computing Environments	POSIX	
Portal		A Web portal is a **Web site** that provides a starting point, gateway, or portal to other resources on the **Internet** or an intranet. Intranet portals are also known as "enterprise information portals" (EIP). Examples of existing portals are Yahoo, Excite, Lycos, Altavista, Infoseek, and Hotbot. (Source: http://en.wikipedia.org/wiki/web_portal)
Portal Page		A complete document rendered by a portal. (Source: http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf)
Portlet		A reusable Web component that displays relevant information to portal users. Examples for portlets include email, weather, discussion forums, and news. The purpose of the **Web Services for Remote Portlets** (WSRP) interface is to provide a **Web services** standard that allows for the "plug-n-play" of **portals**, other intermediary **Web applications** that

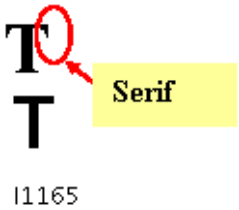
		aggregate content, and applications from disparate sources. The portlet specification enables interoperability between portlets and portals. This specification defines a set of **APIs** for portal computing that addresses the areas of aggregation, personalization, presentation, and security. (Source: http://en.wikipedia.org/wiki/Portlets)
Portlet Container		A portlet container provides a runtime environment for **portlets** implemented according to the portlet **API**. In this environment portlets can be instantiated, used, and finally destroyed. The portlet container is not a standalone container like the **servlet** container; instead it is implemented as a thin layer on top of the servlet container and reuses the functionality provided by the servlet container. (Source: http://portals.apache.org/pluto/)
Portlet Specification	JSR 168	To enable interoperability between **portlets** and **portals**, this specification defines a set of **APIs** for portal computing that address the areas of aggregation, personalization, presentation, and security. (Source: http://www.jcp.org/en/jsr/detail?id=168)
Primary Key	PK	An object that uniquely identifies a row within a table.
Private Key		The private key is one of a pair of keys that are generated as part of asymmetric key cryptography. The private key is kept secret; the **public key** can be shared openly with others.
Protocol		An agreed-upon format for transmitting data between two devices. The protocol determines the type of error checking to be used, data compression method, if any, how the sending device will indicate that it has finished sending a message, and how the receiving device will indicate that it has received a message. (Source: http://www.webopedia.com/TERM/p/protocol.html)
Proxy		A **server** that sits between a client application, such as a **Web browser**, and a real server. It intercepts all requests to the real server to see if it can fulfill the requests itself. If not, it forwards the request to the real server. Proxy servers have two main purposes: improve performance and filter requests. (Source: http://www.webopedia.com/TERM/p/proxy_server.html)
Proxy Pattern		Provides a surrogate or placeholder for another object to control access to it.
Public Key	PK	See **Public Key Cryptography**.
Public Key Certificate		Used in client-certificate authentication to enable the server, and optionally the client, to authenticate each other. The public key certificate is the digital equivalent of a passport. It is issued by a trusted organization, called a certificate authority, and provides identification for the bearer. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)

Public Key Cryptography		Public key cryptography, also known as asymmetric cryptography, is a form of cryptography in which a user has a pair of cryptographic keys - a public key and a private key. The private key is kept secret, while the public key may be widely distributed. The keys are related mathematically, but the private key cannot be practically derived from the public key. A message encrypted with the public key can be decrypted only with the corresponding private key. (Source: http://en.wikipedia.org/wiki/Public_key)
Public Key Enabling	PK-Enabling	The incorporation of the use of certificates for security services such as authentication, confidentiality, data integrity, and nonrepudiation. PK-Enabling involves replacing existing or creating new user authentication systems using certificates instead of other technologies, such as userid and password or **Internet Protocol** filtering; implementing public key technology to digitally sign, in a legally enforceable manner, transactions and documents; or using public key technology, generally in conjunction with standard symmetric encryption technology, to encrypt information at rest and/or in transit. (Source: DoD Instruction 8520.2, *Public Key Infrastructure (PKI) and Public Key (PK) Enabling*, 1 April 2004 [R1206])
Public Key Infrastructure	PKI	Framework established to issue, maintain, and revoke public key certificates accommodating a variety of security technologies, including the use of software. (Source: CNSS Instruction No. 4009, Revised May 2003, *National Information Assurance (IA) Glossary*)
Publish/Subscribe Messaging System		A messaging system in which clients address messages to a specific node in a content hierarchy, called a topic. Publishers and subscribers are generally anonymous and can dynamically publish or subscribe to the content hierarchy. The system takes care of distributing the messages arriving from a node's multiple publishers to its multiple subscribers. Messages are generally not persistent and will only be received by subscribers who are listening at the time the message is sent. A special case known as a "durable subscription" allows subscribers to receive messages sent while the subscribers are not active. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Quality of Service	QoS	Data timeliness, accuracy, completeness, integrity, and ease of use. Refers to the probability of the network meeting a given traffic contract. In many cases is used informally to refer to the probability of a packet passing between two points in the network. (Source: http://en.wikipedia.org/wiki/Quality_of_service) -OR- A defined level of performance that adapts to the environment in which it is operating. QoS may be requested by the user of the information. The level of QoS provided is based on the request, the available capabilities of the provider, and the priority of the user.
Real-Time		An operation within a larger dynamic system is called a real-time operation if the combined reaction- and operation-time of a task is shorter than the maximum delay that is allowed, in view of circumstances outside the operation. The task must also occur before the system to be controlled becomes unstable. A real-time operation is not necessarily fast, as slow systems can allow slow real-time operations. This applies for all types of dynamically changing systems.

		The polar opposite of a real-time operation is a batch job with interactive timesharing falling somewhere in-between the two extremes. (Source: http://en.wikipedia.org/wiki/Real_time)
Real-Time System		A system in which the correctness of system behavior depends on both the logical correctness of the computation and the time at which the result is produced. For a real-time system, the system fails if its timing constraints are not met. "Real time" is not necessarily synonymous with "fast." The latency of the response might not be an issue, and it could be on the order of seconds or minutes. But the bounded latency that is sufficient to solve the problem at hand is guaranteed by the system. "Bounded" means that the response is neither too early nor too late. In real-time systems, early can be as bad as late.
Refactoring		Refactoring is often used to describe modifying source code without changing its external behavior, and is sometimes informally referred to as "cleaning it up." Refactoring is often practiced as part of the software development cycle: developers alternate between adding new tests and functionality and refactoring the code to improve its internal consistency and clarity. Testing ensures that refactoring does not change the behavior of the code.
Reference Data Set		A reference data set is a collection of related data that represent a defined entity within a Community of Interest. Examples of reference data sets include country codes, U.S. state codes, and marital status codes. (Soure: **DoD Metadata Registry and Clearinghouse**; https://metadata.dod.mil/mdr/other.htm?page=help)
Referential Integrity		A feature provided by RDBMSs that prevents users or applications from entering inconsistent data. Most RDBMSs have various referential integrity rules that you can apply when you create a relationship between two tables.
Registered Namespace		A namespace that has been registered and approved with a **namespace** registration services. For the DoD, use the **DoD Metadata Registry**.
Registration Web Service	RWS	**Horizontal Fusion** (HF) **service** used by data producers to register content sources.
Relational Database	RDB	A collection of data items organized as a set of formally-described tables from which data can be accessed or reassembled in many different ways without having to reorganize the database tables.
Relational Database Management System	RDBMS	A database management system (DBMS) that is based on the relational model or that presents the data to the user as relations. A collection of tables, each table consisting of a set of rows and columns, can satisfy this property. RDBMSs also provide relational operators to manipulate the data in tabular form. (Source: http://en.wikipedia.org/wiki/RDBMS)
Relative Font Size		Fonts that display according to the size of the surrounding text. Some designers call them scalable fonts. Instead of displaying a

		fixed pixel size, a relative font size displays as a percentage of the surrounding elements. (Source: http://www.netmechanic.com/news/vol5/design_no13.htm)
Remote Method Invocation	RMI	A technology that allows an object running in one Java virtual machine to invoke methods on an object running in a different Java virtual machine. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Remote Procedure Call	RPC	An alternative to sockets that abstracts the communication interface to the level of a procedure call. The programmer has the illusion of calling a local procedure, but in fact the arguments of the call are packaged and sent to the remove target of the cell. RPC systems encode arguments and return values using an external data representation such as XDR. RPC does not translate well into distributed object systems, which require communication between program-level objects in different address spaces. To match the semantics of object invocation, distributed object systems require RMI. A local surrogate (stub) object manages the invocation on a remote object.
Resource Definition Framework	RDF	
Role-Based Access Control	RBAC	With RBAC, security is managed at a level that corresponds closely to the organization's structure. Each user is assigned one or more roles, and each role is assigned one or more privileges that are permitted to users in that role. Security administration with RBAC consists of determining the operations that must be executed by persons in particular jobs, and assigning employees to the proper roles. Complexities introduced by mutually exclusive roles or role hierarchies are handled by the RBAC software, making security administration easier. (Source: National Institute of Standards and Technology Computer Security Resource Center, http://csrc.nist.gov/groups/SNS/rbac/)
Rollback		The point in a transaction when all updates to any resources involved in the transaction are reversed. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Router		A device that forwards data packets along networks. A router is connected to at least two networks, commonly two **local area networks** (LANs) or wide area networks (WANs) or a LAN and its Internet Service Provider's network. Routers are located at gateways, the places where two or more networks connect. (Source: http://www.webopedia.com/TERM/r/router.html)
Sans Serif Font		A sans serif font is a font that has no serifs. Examples are ***Arial***, ***Century Gothic***, and ***Helvetica***. (Source: http://web.mit.edu/abiword_v2.0.10/Tutorials/klw/glossary.html)
SCA Operating Environment	OE	**SCA** Operating Environment: The SCA OE describes the requirements of the operating system, middleware, and the CF interfaces and operations.

Schema		A diagrammatic representation, an outline, or a model. In relation to data management, a schema can represent any generic model or structure that deals with the organization, format, structure, or relationship of data. Some examples of schemas are (1) a database table and relational structure, (2) a **document type definition** (DTD), (3) a data structure used to pass information between systems, and (4) an **XML schema document** (XSD) that represents a data structure and related information encoded as XML. Schemas typically do not contain information specific to a particular instance of data (Source: DoD 8320.02-G, 12 April 2006, *Guidance for Implementing Net-Centric Data Sharing*)
Search Web Service	SWS	**Horizontal Fusion** (HF) **service** used to search for content from registered sources.
Secret Internet Protocol Router Network	SIPRNet	SIPRNet is DoD's largest interoperable command and control data network, supporting the **Global Command and Control System** (GCCS), the Defense Message System (DMS), collaborative planning and numerous other classified warfighter applications. Direct connection data rates range from 56 kbps to 155 Mbps. Remote dial-up services are available up to 19.2 kbps. (Source: http://www.disa.mil/services/data.html)
Secret Key		The asymmetric key cryptography approach generates two keys, a public key and a private key. The **private key** is often referred to as the secret key.
Secure Hash Algorithm	SHA	The SHA (Secure Hash Algorithm) family is a set of related cryptographic hash functions. In cryptography, a cryptographic hash function is a hash function with certain additional security properties to make it suitable for use as a primitive in various information security applications, such as authentication and message integrity. A hash function takes a long string (or message) of any length as input and produces a fixed length string as output, sometimes termed a message **digest** or a digital fingerprint. (Source: http://en.wikipedia.org/wiki/SHA#SHA-0_and_SHA-1)
Secure Sockets Layer	SSL	A protocol for transmitting private documents via the Internet. SSL uses a cryptographic system employing two keys to encrypt data: a **public key** known to everyone and a **private** or **secret key** known only to the recipient of the message. (Source: http://www.webopedia.com/TERM/S/SSL.html)
Security Assertion Markup Language	SAML	An XML standard for exchanging authentication and authorization data between security domains; that is, between an identity provider and a service provider. SAML is a product of the **OASIS** Security Services Technical Committee. (Source: http://en.wikipedia.org/wiki/SAML)
Security Technical Implementation Guide	STIG	Configuration standards for DoD **IA** and IA-enabled devices/systems. (Source: http://iase.disa.mil/stigs/index.html)
Semantics		The implied meaning of data, the study or words and their meanings.

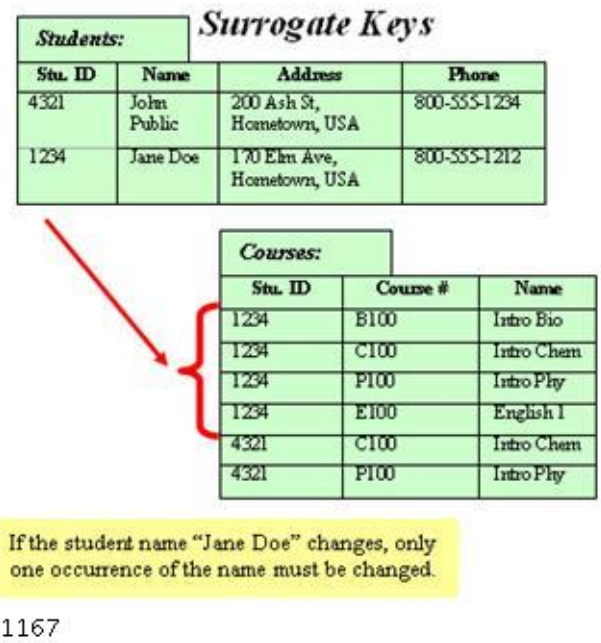
Sensitive Compartmented Information	SCI	Classified information concerning or derived from intelligence sources, methods, or analytical processes, that is required to be handled within formal access control systems established by the Director of Central Intelligence (DCI). (Source: DoDD 8520.1, 20 December 2001, *Protection of Sensitive Compartmented Information (SCI)*, Page 2, Section 3.3)
Serialization		Serialization is the process of writing a complex object into a serial stream of data. When the data is successfully transferred, the data can be **deserialized** back into a complex object. ***Note:*** *The process of transferring data using serialization and* ***deserialization*** *is called* ***marshalling****.*
Serif Font		A serif is a feature of the letters in a given typeset. They appear at the end of lines within the letters. An example would be the letter T in Times New Roman - at the end of each horizontal line is a tick that hangs down (that is the serif). Serif fonts include ***Times New Roman***, ***Bookman Oldstyle***, and ***Courier***. 
Server		A computer software application that carries out some task (i.e., provides a service) on behalf of yet another piece of software called a **client**.
Service		A service is an autonomous encapsulation of some business or mission functionality. The service concept includes the notion of service providers and service consumers interacting via well-defined reusable interfaces. ***Note:*** *See the Service-Oriented Architecture [P1304] perspective in Part 1 for additional information concerning services including implementation characteristics.*
Service Access Point	SAP	A SAP provides all of the information necessary for a user to access and consume a service including the logical and physical location of the service on the net.
Service Definition Framework	SDF	An SDF provides a common frame of reference for service users, customers, developers, providers, and managers. Its structure and methodology enable full definition of the **Service Access Points** (SAPs) for a service.

		Note: *See P1296 [P1296]:* ***Service Definition Framework*** *for additional information.*
Service Discovery	SD	Provides a ***yellow pages***, categorized by **DoD** function, enabling users to advertise and locate capabilities available on the network.
Service Level Agreement	SLA	A contractual vehicle between a service provider and a service consumer. It specifies performance requirements, measures of effectiveness, reporting, cost, and recourse. It usually defines repair turnaround times for users.
Service Management		Enables monitoring of DoD **Web services**. Provides reporting of service-level information to potential and current service consumers, program analysts, and program managers.
Service-Oriented Architecture	SOA	NESI describes SOA as an architectural style used to design, develop, and deploy information technology (IT) systems based on decomposing functionality into services with well-defined interfaces. ***Note:*** *See the Service-Oriented Architecture [P1304] perspective in Part 1 for additional information.*
Service Provider		The person, organization, or automated asset that implements and operates a service.
Service Registry		Provides descriptive information about a service, enabling the lookup and discovery of services.
Servlet		A Java program that extends the functionality of a Web server, generating dynamic content and interacting with Web applications using a request-response paradigm. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Session		An interaction between system entities of finite duration, often involving a user, typified by the maintenance of some state of the interaction for the duration of the interaction. (Source: http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf)
Session Key		A session key is an encryption and decryption key randomly generated to ensure the security of a communications session between a user and a computer or between two computers. Session keys are sometimes called symmetric keys, because the same key is used for both encryption and decryption. Throughout each session, the key is transmitted with each message and is encrypted with the recipient's public key. Because much of their security relies upon the brevity of their use, session keys are often changed frequently.

Simple Mail Transfer Protocol	SMTP	
Simple Structured Data		Simple Structured Data has an uncomplicated data structure. All requisite **metadata** is provided and simple data types only are used (e.g., integers, long integers, strings, and simple lists.
Simple Unstructured Data		Simple Unstructured Data has uncomplicated data structure but not all requisite **metadata** is provided.
Single Sign-On	SSO	
Single Touch Point		The portal becomes the delivery mechanism for all business information services.
Situation Awareness Data Link	SADL	An Enhanced Position Location and Reporting System (EPLRS) radio modified for use in an aircraft. SADL and EPLRS radios are used to establish a common secure tactical data link network. (Source: http://aatc.aztucs.ang.af.mil/aatcinfo.htm)
Smart Card		A credit card-size device, normally for carrying and use by personnel, that contains one or more integrated circuits and also may employ one or more of the following technologies: magnetic stripe, bar codes (linear and two-dimensional), non-contact and radio frequency transmitters, biometric information, encryption and authentication, or photo identification. (Source: DoDD 8190.3, *Smart Card Technology*, 31 August 2003, Page 2, Section 3.2)
SOAP		SOAP Version 1.2 is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics. (Source: SOAP Version 1.2 Second Edition, http://www.w3.org/TR/soap12-part1/#intro) ***Note:*** *The World Wide Web Consortium (W3C) changed the name of this protocol from* ***Simple Object Access Protocol 1.1 (SOAP)*** *to* ***SOAP Version 1.2*** *in the current version.*
Software Communications Architecture	SCA	An implementation-independent framework for the development of software for an established hardware platform, such as software defined radios.
Software Component		A software component is a software system element offering a predefined service and able to communicate with other components. It is a unit of independent deployment and versioning, encapsulated, multiple-use, non-context-specific and composeable with other components. Source: http://en.wikipedia.org/wiki/Software_component#Software_component

Software Developers Kit	SDK	A set of development tools that allows a software engineer to create applications for a certain software package, software framework, hardware platform, computer system, operating system, and so on. It may be as simple as an application programming interface in the form of some files to interface to a particular programming language, or as complex as sophisticated hardware to communicate with a certain embedded system. Common tools include debugging aids and other utilities. SDKs frequently include sample code, technical notes, and other supporting documentation to clarify points from the primary reference material. (Source: http://en.wikipedia.org/wiki/SDK)
Spyware		Any software that covertly gathers user information through the user's **Internet** connection without the user's knowledge, usually for advertising purposes. (Source: http://www.webopedia.com/TERM/s/spyware.html)
Storage		Provides physical and virtual places to host and retain data for purposes such as content staging, continuity of operations, or archival.
Stored Procedure		A unit or module of code that executes in a database and implement some bit of application logic or business rule. Often written in proprietary language such as Oracle's PL/SQL or Sybase's Transact-SQL.
Stovepipe System		A stovepipe system is a legacy system that is an assemblage of inter-related elements that are so tightly bound together that the individual elements cannot be differentiated, upgraded or refactored. The stovepipe system must be maintained until it can be entirely replaced by a new system. Examples of stovepipe systems: • Systems for which new hardware is no longer available • Systems whose original source code has been lost • Systems that were built using old or ad hoc engineering methodologies for which support can no longer be found The term is also used to describe a system that does not interoperate with other systems, presuming instead that it is the only extant system. A stovepipe system is an example of an anti-pattern legacy system and demonstrates software brittleness. (Source: http://en.wikipedia.org/wiki/Stovepipe_system)
Structured Identifier		Identifiers are labels which serve as references to the identity of resources, assets, **nodes**, **components**, and other entities. Ideally, identifiers should quickly answer at least one of the following common questions about the entity: who, what, where, when and which. Identifiers include, for example, names (for user environment usage), addresses (for transport usage), pathnames (for computing infrastructure usage), cryptographic keys (for security/**IA** usage) and above all, **Uniform Resource Identifiers** or **URIs** (for management, applications and services).

		Not all identifiers are structured; however, a benefit of structured identifiers is that they are useful for component software and hardware to understand and parse progressively the data expressed within the identifier. Progressive understanding of a standardized structured identifier is a form of negotiation that enables different entities either to interoperate correctly or to conclude efficiently that interoperation is not possible, even when the entities have never communicated before. For example, structured identifiers commonly identify the type and instance of an entity. Structuring an identifier into type portions and instance portions enables it to answer quickly and efficiently both what type of interactions are possible and with which instance of that type. Another common practice is for structured identifiers to express the hierarchical relationship between entities. Examples of structured identifiers expressing a hierarchical relationship include domain names such as **`nesipublic.spawar.navy.mil`** or the familiar telephone number hierarchy of country code, area code, exchange and line. The hierarchical structure in those cases indicates that there is a governance authority hierarchy whose top level delegates authority to the lower ones. Examples of useful standards for interoperable net-centric structured identifiers include the following: • (**IETF**) Request for Comments (RFC) 3986, *Uniform Resource Identifier: Generic Syntax*, http://tools.ietf.org/html/rfc3986 • IETF RFC 1035, *Domain Names - Implementation and Specification*, http://tools.ietf.org/html/rfc1035 • *Multi-Purpose Internet Mail Extensions (MIME) Media Types*, http://www.iana.org/assignments/media-types/ • *XML Path Language (XPath) Version 1.0*, http://www.w3.org/TR/xpath
Structured Query Language	SQL	The standardized relational database language for defining database objects and manipulating data. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Structured Query Language 1992	SQL-92	The SQL-92 and SQL:1999 standards are very detailed and specific. At the current time, no **RDBMS** vendors fully support the entire standard. Vendors that claim they are SQL-92-compliant or SQL:1999-compliant are actually only compliant to a certain level. The SQL-92 standard defines the following levels, which also apply to SQL:1999: (1) Notational; (2) Transitional level SQL92; (3) Intermediate level SQL92; (4) .Full SQL92. (Source:http://dbs.uni-leipzig.de/en/lokal/standards.pdf; http://developer.mimer.com/documentation/html_82/Mimer_SQL_Reference_Manual/Intro_SQL_Stds3.html)
Structured Query Language 1999	SQL-99	See **SQL-92**.
Style Sheet		Style sheets describe how documents are presented on screens, in print, or perhaps how they are pronounced. (Source: http://www.w3.org/Style)
Surrogate Key		A surrogate key is a primary key that has been explicitly created and has no relationship with the naturally occurring data found within a table.

		 I1167 See **Natural Key** and **Primary Key**.
Sustainment		One of the two major efforts (with disposal) of the Operations and Support phase of a DoD acquisition program. Sustainment includes supply, maintenance, transportation, sustaining engineering, data management, configuration management, manpower, personnel, training, habitability, survivability, environment, safety (including explosives safety), occupational health, protection of critical program information, anti-tamper provisions, and **Information Technology** (IT), including **National Security Systems** (NSS), supportability and interoperability functions. (Source: DoD Instruction 5000.2, 12 May 2003, *Operation of the Defense Acquisition System*, Section 3.9.2)
Symmetric Key Algorithm		Encryption algorithm where the same key is used for both encrypting and decrypting a message.
System		A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected (Rechtin, 2000). (Source: International Council on Systems Enginering, *A consensus of the INCOSE Fellows*, http://www.incose.org/practice/fellowsconsensus.aspx)
System Component		A basic part of a system. System components may be personnel, hardware, software, facilities, data, material, services, and/or techniques that satisfy one or more requirements in the lowest levels of the functional architecture. System components may be subsystems and/or configuration items.

		Note: *See* ***component****.*
Systems and Services View	SV	The SV is a set of graphical and textual products that describes systems and interconnections providing for, or supporting, DoD functions. DoD functions include both warfighting and business functions. The SV associates systems resources to the Operational View (OV). These systems resources support the operational activities and facilitate the exchange of information among operational nodes. (Source: DoDAF v1.5 Volume I: Definitions and Guidelines, 23 April 2007)
Taxonomy		The science of categorization, or classification, of things based on a predetermined system. In reference to Web sites and portals, a site's taxonomy is the way it organizes its data into categories and subcategories, sometimes displayed in a site map. (Source: http://www.webopedia.com/TERM/t/taxonomy.html)
Taxonomy Gallery		The Taxonomy Gallery [of the **DoD Metadata Registry and Clearinghouse**] provides XML-based **taxonomy** files that describe one or more nodes in a hierarchical classification of items, and their relationships to other nodes. The taxonomy files registered with the Taxonomy Gallery are organized by governance namespace. (Source: http://www.disa.mil/nces/development/developer_doc_overview.html)
Technical Standards View	TV	The TV is the minimal set of rules governing the arrangement, interaction, and interdependence of system parts or elements. Its purpose is to ensure that a system satisfies a specified set of operational requirements. The TV provides the technical systems implementation guidelines upon which engineering specifications are based, common building blocks are established, and product lines are developed. The TV includes a collection of the technical standards, implementation conventions, standards options, rules, and criteria organized into profile(s) that govern systems and system elements for a given architecture. (Source: *DoDAF* v1.5 Volume 1: Definitions and Guidelines, 23 April 2007)
Tenet		Net-centric design precept.
Test and Evaluation Master Plan	TEMP	Describes all planned testing, including measures to evaluate the performance of the system during test periods, an integrated test schedule, and resource requirements.
Topic		Topics are used to manage content flow between publishers and subscribers. Topics must be known in such a way that subscribers can refer to them unambiguously. In **DDS**, Topics conceptually fits between **publications** and **subscriptions** and associate a name (unique in the **domain**), a data-type, and **QoS** parameters related to the data.
Transaction		A set of input data that triggers execution of a specific processor job. Usually manipulates data that may need to be rolled back to the original values if any part of the transaction fails. Transactions enable

		multiple users to access the same data concurrently. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Transmission Control Protocol	TCP	One of the core protocols of the Internet protocol suite. Using TCP, programs on networked computers can create connections to one another, over which they can send data. The protocol guarantees that data sent by one endpoint will be received in the same order by the other, without any pieces missing. It also distinguishes data for different applications (such as a Web server and an email server) on the same computer. (Source: http://en.wikipedia.org/wiki/Transmission_Control_Protocol)
Transmission Control Protocol/Internet Protocol	TCP/IP	A suite of communications protocols used to connect hosts on the Internet. TCP/IP uses several protocols, the two main ones being TCP and IP. TCP/IP is built into the UNIX operating system and is used by the Internet, making it the de facto standard for transmitting data over networks. Even network operating systems that have their own protocols, such as Netware, also support TCP/IP.
Transport Layer Security	TLS	A protocol that guarantees privacy and data integrity between client/server applications communicating over the Internet. The TLS protocol is made up of two layers: • The TLS Record Protocol -- layered on top of a reliable transport protocol, such as TCP, it ensures that the connection is private by using symmetric data encryption and it ensures that the connection is reliable. The TLS Record Protocol also is used for encapsulation of higher-level protocols, such as the TLS Handshake Protocol. • The TLS Handshake Protocol -- allows authentication between the server and client and the negotiation of an encryption algorithm and cryptographic keys before the application protocol transmits or receives any data. (Source: http://www.webopedia.com/TERM/T/TLS.html)
Trigger		In a DBMS, a trigger is a SQL procedure that initiates (fires) an action when an event (INSERT, DELETE, or UPDATE) occurs. Since triggers are event-driven specialized procedures, the DBMS stores and manages them. A trigger cannot be called or executed; the DBMS automatically fires the trigger as a result of a data modification to the associated table. Triggers maintain the referential integrity of data by changing the data in a systematic fashion.
Triple Data Encryption Algorithm	TDEA	An encryption algorithm whose key consists of three DES (Data Encryption Standard) keys, which is also referred to as a key bundle. A DES key consists of 64 binary digits ("0"s or "1"s) of which 56 bits are randomly generated and used directly by the algorithm. (The other 8 bits, which are not used by the algorithm, may be used for error detection.) Each TDEA encryption/decryption operation (as specified in ANSI X9.52) is a compound operation of DES encryption and decryption operations. Let EK(I) and DK(I) represent the DES encryption and decryption of ***I*** using DES key ***K*** respectively. (Source: http://www.atis.org/tg2k/_triple_data_encryption_algorithm.html)

Trusted Guard		Accredited to pass information between two networks at different security levels according to well defined rules and other controls. Guard products only pass defined types of information (e.g., email, images, or formatted messages). A key challenge is how to implement net-centric operations across trusted guards in the presence of **CES** services.
Trusted Platform Module	TPM	The TPM is a microcontroller that stores keys, passwords and digital certificates. It typically is affixed to the motherboard of computers. It potentially can be used in any computing device that requires these functions. The nature of this hardware chip ensures that the information stored there is made more secure from external software attack and physical theft. The TPM standard is a product of the Trusted Computing Group consortium. Source: Encryption of Sensitive Unclassified Data at Test on Mobile Computing Devices and Removable Storage Media R1330: DoD Memorandum , **Encryption of Sensitive Unclassified Data at Rest on Mobile Computing Devices and Removable Storage Media** Chief Information Officer . [http://iase.disa.mil/policy-guidance/dod-dar-tpm-decree07-03-07.pdf]
Trust Point		A trust point is a **Certificate Authority** (**CA**) that is the root of all trust for all CAs in a CA hierarchy.
Tunneling		Transporting IPv6 traffic through IPv4 networks by encapsulating IPv6 packet in IPv4 and vice-versa.
Unclassified but Sensitive Internet Protocol Router Network	NIPRNet	NIPRNet provides seamless interoperability for unclassified combat support applications, as well as controlled access to the Internet. Direct connection data rates range from 56Kbps to 622Mbps. Remote dial-up services are available up to 56Kbps. (Source: http://www.disa.mil/main/prodsol/data.html)
Unicode		A standard defined by the Unicode Consortium. Unicode uses a 16-bit code page that maps digits to characters in languages around the world. Because 16 bits covers 32,768 codes, Unicode is large enough to include all the world's languages, with the exception of ideographic languages that have a different character for every concept, such as Chinese. For more information, see http://www.unicode.org/. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Unified Class Library		With the introduction of **.NET**, Microsoft redesigned the access to common system components and services such as XML Web services, Enterprise Services, ADO.NET, and XML by creating a single object-oriented library. All the Microsoft Visual .NET languages (Visual Basic, C++, J#, C#, etc.) have access to this library. To make access to these objects available within the various languages, Microsoft provided infrastructure such as hierarchical namespaces, structures, types, and common objects like collections.
Unified Modeling Language	UML	In the field of software engineering, the Unified Modeling Language (UML) is a standardized specification language for object modeling. UML is a general-purpose modeling language that includes a

		graphical notation used to create an abstract model of a system, referred to as a UML model. UML is officially defined at the **Object Management Group** (OMG) by the UML metamodel, a Meta-Object Facility metamodel (MOF). (Source: http://en.wikipedia.org/wiki/Unified_Modeling_Language; 30 March 2007)
Uniform Resource Identifier	URI	An encoded address that represents any Web resource, such as an **HTML** document, image, video clip, or program. As opposed to a **URL** or a **URN**, which are concrete entities, a URI is an abstract superclass. (Source: http://publib.boulder.ibm.com/infocenter/adiehelp/index.jsp?topic=/com.ibm.wsinted.glossary.doc/topics/glossary.html)
Uniform Resource Locator	URL	A sequence of characters that represents information resources on a computer or in a network such as the Internet. This sequence of characters includes (1) the abbreviated name of the protocol used to access the information resource and (2) the information used by the protocol to locate the information resource.(Source: http://publib.boulder.ibm.com/infocenter/adiehelp/index.jsp?topic=/com.ibm.wsinted.glossary.doc/topics/glossary.html)
Uniform Resource Name	URN	A name that uniquely identifies a **Web service** to a **client**. (Source: http://publib.boulder.ibm.com/infocenter/adiehelp/index.jsp?topic=/com.ibm.wsinted.glossary.doc/topics/glossary.html)
UNIQUE Key Integrity Constraint		A **`UNIQUE`** key integrity constraint requires that every value in a column or set of columns (key) be unique; that is, no two rows of a table have duplicate values in a specified column or set of columns. (Source: http://www.lc.leidenuniv.nl/awcourse/oracle/server.920/a96524/c22integ.htm)
Universal Description, Discovery, and Integration	UDDI	An industry initiative to create a platform-independent, open framework for describing services, discovering businesses, and integrating business services using the Internet, as well as a registry. It is being developed by a vendor consortium. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Use-Case		A sequence of actions, performed by a system, that yields a result of value to a user. A set of actions, including variants, that a system performs that yields an observable result of value to a particular actor.
User Datagram Protocol	UDP	A connectionless protocol that, like **TCP**, runs on top of **Internet Protocol** (IP) networks. Unlike **Transmission Control Protocol/Internet Protocol** (TCP/IP), UDP/IP provides very few error recovery services, offering instead a direct way to send and receive datagrams over an IP network. It's used primarily for broadcasting messages over a network. (Source: http://www.webopedia.com/TERM/U/User_Datagram_Protocol.html)
Valid		A valid XML document has data that conforms to a particular set of user-defined content rules, or XML Schemas, that describe correct data values and locations. For example, if an element in a document is required to contain text that can be interpreted as being an integer numeric value, and it instead has the text ***hello***, is empty, or has other

		elements in its content, then the document is not valid. (Source: adapted from http://en.wikipedia.org/wiki/XML; 9/11/2006)
Virtual Private Network	VPN	A network that is constructed by using public wires to connect nodes. For example, there are a number of systems that enable the creation of networks using the Internet as the medium for transporting data. These systems use encryption and other security mechanisms to ensure that only authorized users can access the network and that the data cannot be intercepted. (Source: http://www.webopedia.com/TERM/V/VPN.html)
Visual Basic Scripting	VBScript	VBScript (Visual Basic Scripting) is a programming language developed by Microsoft which is similar to **JavaScript**. It is used to embed code into **HTML** pages. It is actually a subset of Microsoft's Visual Basic. (Source: Strategic Web Ventures Glossary, http://www.strategicwebventures.com/definitions/Glossary/VBScript)
VoiceXML	VXML	VoiceXML (VXML) is the **W3C** standard **XML** format for specifying interactive voice dialogues between a human and a computer. It is fully analogous to **HTML**, and brings the same advantages of **Web application** development and deployment to voice applications that HTML brings to visual applications. Just as HTML documents are interpreted by a visual web browser, VoiceXML documents are interpreted by a voice browser. A common architecture is to deploy banks of voice browsers attached to the public switched telephone network (PSTN) so that users can simply pick up a phone to interact with voice applications. VoiceXML has tags that instruct the voice browser to provide speech synthesis, automatic speech recognition, dialog management, and soundfile playback.
Web Application		A collection of components that can be bundled together and run in multiple containers from multiple vendors. -OR- An application written for the Internet, including those built with Java technologies such as Java Server Pages and servlets, and those built with non-Java technologies such as CGI and Perl. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Web Application Archive	WAR	A **JAR** archive that contains a **Web module**. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Web Browser		A client program that initiates requests to a **Web server** and displays the information that the server returns. (Source: http://publib.boulder.ibm.com/infocenter/adiehelp/index.jsp?topic=/com.ibm.wsinted.glossary.doc/topics/glossary.html)
Web Container		A container that implements the Web-component contract of the **J2EE** architecture. This contract specifies a runtime environment for Web components that includes security, concurrency, life-cycle management, transaction, deployment, and other services. A Web container provides the same services as a **JSP** container as well as a federated view of the J2EE platform **APIs**. A Web container is provided by a Web or J2EE server. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)

Web Module		A deployable unit that consists of one or more Web components, other resources, and a Web application deployment descriptor. The Web module is contained in a hierarchy of directories and files in a standard Web application format. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Web Ontology Language	OWL	A markup language for publishing and sharing data using ontologies on the Internet. (Source: http://en.wikipedia.org/wiki/Web_Ontology_Language)
Web Page		A document created with **HTML** (HyperText Markup Language) that is part of a group of hypertext documents or resources available on the World Wide Web. Collectively, these documents and resources form what is known as a **Web site**. You can read HTML documents that reside somewhere on the Internet or on your local hard drive with software called a **Web browser**. Web pages can contain hypertext links to other places within the same document, to other documents at the same Web site, or to documents at other Web sites.
Web Server		Software that provides services to access the Internet, an intranet, or an extranet. A Web server hosts **Web sites**, provides support for HTTP and other protocols, and executes server-side programs (such as **CGI** scripts or servlets) that perform certain functions. In the **J2EE** architecture, a Web server provides services to a **Web container**. For example, a Web container typically relies on a Web server to provide **HTTP** message handling. The J2EE architecture assumes that a Web container is hosted by a Web server from the same vendor, so it does not specify the contract between these two entities. A Web server can host one or more Web containers. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
Web Service		A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically **WSDL**). Other systems interact with the Web service in a manner prescribed by its description using **SOAP** messages, typically conveyed using **HTTP** with an **XML** serialization in conjunction with other Web-related standards. (Source: http://www.w3.org/TR/ws-gloss/)
Web Services Description Language	WSDL	WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. (Source: W3C Note on WSDL 1.1 of 15 March 2001 http://www.w3.org/TR/wsdl)
Web Services for Interactive Applications	WSIA	
Web Services for Remote Portlets	WSRP	The WSRP specification defines a **Web service** interface for interacting with interactive presentation-oriented Web services. It has been produced through the joint efforts of the Web Services for Interactive

		Applications (**WSIA**) and Web Services for Remote Portals (WSRP) OASIS Technical Committees. Scenarios that motivate WSRP/ WSIA functionality include (1) **portal** servers providing **portlets** as presentation-oriented Web services that can be used by aggregation engines; (2) portal servers consuming presentation-oriented Web services provided by portal or non-portal content providers and integrating them into a portal framework. (Source: http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf)
Web Services Interoperability Organization	WS-I	WS-I is an open industry organization chartered to promote Web services interoperability across platforms, operating systems and programming languages. The organization's diverse community of Web services leaders helps customers to develop interoperable Web services by providing guidance, recommended practices and supporting resources. (Source: http://www.ws-i.org/about/Default.aspx)
Web Site		A Web site, website, or WWW site (often shortened to just "site") is a collection of Web pages (i.e., HTML/XHTML documents accessible via **HTTP** on the Internet). All publicly accessible Web sites in existence comprise the World Wide Web. The pages of a Web site are accessed from a common root URL, the homepage, and usually reside on the same physical server. The URLs of the pages organize them into a hierarchy, although the hyperlinks between them control how the reader perceives the overall structure and how the traffic flows between the different parts of the site. (Source: http://en.wikipedia.org/wiki/web_site)
Well-Formed		A textual object is a well-formed **XML document** if: 1. Taken as a whole, it matches the production labeled document. 2. It meets all the well-formedness constraints given in this specification. 3. Each of the parsed entities which is referenced directly or indirectly within the document is well-formed. (Source: http://www.w3.org/TR/REC-xml/#dt-wellformed)
Wireless Application Protocol	WAP	WAP is an open international standard for applications that use wireless communication, such as Internet access from a mobile phone. WAP provides services equivalent to a Web browser with some mobile-specific additions. It is specifically designed to address the limitations of very small portable devices. During its first years of existence WAP suffered from considerable negative media attention and has been criticised heavily for its design choices and limitations. (Source: http://en.wikipedia.org/wiki/WAP)
Wireless Markup Language	WML	WML is the primary content format for devices that implement the **WAP** (Wireless Application Protocol) specification based on XML, such as mobile phones. (Source: http://en.wikipedia.org/wiki/Wireless_Markup_Language)
Wire Protocol		In a network, it is the mechanism for transmitting data from point a. to point b. It often refers to a distributed object protocol such as Remote Method Invocation, or **RMI**, which is software only and which

		invokes the running of programs on remote servers. (Source: http://www.techweb.com/encyclopedia/defineterm.jhtml?term=wire+protocol)
Wisdom		Knowledge with information so thoroughly assimilated as to have produced sagacity, judgment, and insight. The ability to use knowledge for a purpose.
World Wide Web	WWW	The World Wide Web ("WWW," or simply "Web") is an information space in which items of interest, referred to as resources, are identified by global identifiers called Uniform Resource Identifiers (**URI**). The term is often mistakenly used as a synonym for the **Internet**, but the web is actually a service that operates over the Internet. (Source: http://en.wikipedia.org/wiki/World_Wide_web)
World Wide Web Consortium	W3C	The World Wide Web Consortium (W3C) is an international consortium where Member organizations, a full-time staff, and the public work together to develop Web standards. W3C's mission is to lead the World Wide Web to its full potential by developing protocols and guidelines that ensure long-term growth for the Web. (Source: http://www.w3.org/Consortium/)
XML Attribute		An XML structural construct. A name-value pair, separated by an equals sign, included inside a tagged element that modifies certain features of the element. All attribute values, including things like size and width, are in fact text strings and not numbers. For XML, all values must be enclosed in quotation marks. Attributes can be declared for an XML element type using an attribute list declaration. (Source: http://msdn2.microsoft.com/en-us/library/ms256452.aspx)
XML Document		A document object that is **well-formed**, according to the XML recommendation, and that might (or might not) be valid. The XML document has a logical structure (composed of declarations, elements, comments, character references, and processing instructions) and a physical structure (composed of entities, starting with the root, or document entity). (Source: http://msdn2.microsoft.com/en-us/library/ms256452.aspx)
XML Element		An XML structural construct. An XML element consists of a start tag, an end tag, and the information between the tags, which is often referred to as the contents. Each element has a type, identified by name, sometimes called its "generic identifier" (GI), and may have a set of attribute specifications. Each attribute specification has a name and a value. An instance of an element is declared using <element> tags. Elements used in an XML file are described by a **DTD** or schema, either of which can provide a description of the structure of the data. (Source: http://msdn2.microsoft.com/en-us/library/ms256452.aspx)
XML Gallery		The XML Gallery [of the **DoD Metadata Registry and Clearinghouse**] contains information resources such as submission packages, elements, attributes, and schemas that have been registered by DOD software developers. These information resources use XML, a platform and vendor independent format for exchanging data, to handle data,

		data structures, and data descriptions (metadata). (Source: http://www.disa.mil/nces/development/developer_doc_overview.html)
XML Information Resources		Document Type Definition (**DTD**) or XML Schema Documents (**XSD**) files.
XML Instance Document		An XML document defined by an XML Schema but is populated with the data, not the definition of the data.
XML Path Language	XPath	The result of an effort to provide a common syntax and semantics for functionality shared between XSL Transformations (XSLT) and XML Pointer Language (XPointer) . The primary purpose of XPath is to address parts of an XML document. It also provides basic facilities for manipulation of strings, numbers, and Booleans. XPath uses a compact, non-XML syntax to facilitate use of XPath within URIs and XML attribute values. XPath gets its name from its use of a path notation as used in URLs for navigating through the hierarchical structure of an XML document. (Source: http://msdn2.microsoft.com/en-us/library/ms256452.aspx)
XML Process Definition Language	XPDL	Is the language proposed by the Workflow Management Coalition (WfMC) to interchange process definitions between different workflow products. The goal of XPDL is to provide a Lingua Franca for the workflow domain allowing for the import and export process definitions between a variety of tools ranging from workflow management systems to modeling and simulation tools.
XML Schema		A database-inspired method for specifying constraints on documents using an XML-based language. Schemas address deficiencies in **DTDs**, such as the inability to constrain the kinds of data that can occur in a particular field. Because schemas are founded on XML, they are hierarchical. Thus it is easier to create an unambiguous specification, and it is possible to determine the scope over which a comment is meant to apply. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
XML Schema Definition	XSD	A language proposed by the **W3C** XML Schema Working Group for use in defining schemas. Schemas are useful for enforcing structure and/or constraining the types of data that can be used validly within other XML documents. XML Schema Definition refers to the fully specified and currently recommended standard for use in authoring XML schemas. Because the XSD specification was only recently finalized, support for it was only made available with the release of MSXML 4.0. It carries out the same basic tasks as DTD, but with more power and flexibility. Unlike DTD, which requires its own language and syntax, XSD uses XML syntax for its language. XSD closely resembles and extends the capabilities of XDR. Unlike XDR, which was implemented and made available by Microsoft in MSXML 2.0 and later releases, the W3C now recommends the use of XSD as a standard for defining XML schemas. (Source: http://msdn2.microsoft.com/en-us/library/ms256452.aspx)
XSL Transformations	XSLT	A language to express the transformation of XML documents into other XML documents. (Source: W3C Glossary)

References

R1001	DOM # a Whatis.com definition - http://whatis.techtarget.com/definition/0,,sid9_gci213910,00.html.
R1002	Latest **SOAP** Versions, http://www.w3.org/TR/soap/
R1003	**Web Service Definition Language** (**WSDL**) - http://www.w3.org/TR/wsdl
R1004	XSD - a Whatis.com definition - http://searchwebservices.techtarget.com/sDefinition/0%2c%2csid26_gci831325%2c00.html
R1007	XSL - definition of XSL in Encyclopedia - http://encyclopedia.laborlawtalk.com/XSL
R1008	OASIS , **Web Services Security: SOAP Message Security 1.1 (WS-Security 2004 [commonly known as WS-Security])** . [http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf]
R1011	ActiveX Control definition - http://isp.webopedia.com/TERM/A/ActiveX_control.html
R1012	Component Object Model definition - http://isp.webopedia.com/TERM/C/Component_Object_Model.html
R1013	JScript definition - http://isp.webopedia.com/TERM/J/JScript.html
R1014	IIS - definition of IIS in Encyclopedia - http://encyclopedia.laborlawtalk.com/IIS
R1015	Personal Web Server - a Whatis.com definition - http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci296469,00.html
R1016	HTML - a Whatis.com definition - http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci212286,00.html
R1017	VBScript definition from The Glossary of Internet Terms - http://www.strategicwebventures.com/definitions/Glossary/VBScript/
R1018	HTML - a Whatis.com definition - http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci212286,00.html
R1019	What is JSP? - http://www.webopedia.com/TERM/J/JSP.html
R1023	For answers to frequently asked questions about **cascading style sheets**, see http://www.blooberry.com/indexdot/css/topics/stylefaq.htm
R1024	Don't Make Me Think by Steve Krug (ISBN 0-7897-2310-7)
R1025	Creating Killer Interactive Web Sites by Adjacency (ISBN 1-56830-373-4)
R1026	Designing Web Usability by Jakob Nielsen (ISBN 1-56205-810-X)
R1027	OMG - http://www.omg.org/gettingstarted/gettingstartedindex.htm
R1031	Adapter pattern - http://c2.com/cgi/wiki?AdapterPattern

R1032	Design patterns: Proxy - http://www.dofactory.com/Patterns/PatternProxy.aspx
R1033	Facade pattern - http://c2.com/cgi/wiki?FacadePattern
R1040	JavaBeans definition - http://isp.webopedia.com/TERM/J/JavaBeans.html
R1041	Model-view-controller - a Whatis.com definition - http://whatis.techtarget.com/definition/0,,sid9_gci214607,00.html
R1042	Java Servlets definition - http://www.fromallangles.com/glossary/web-hosting/terms/java-servlets.htm
R1046	"The Semantic Web," Michael C, Daconata, Leo J. Obrst, Kevin T. Smith; Wiley Publishing Inc., 2003
R1047	http://members.optusnet.com.au/~webindexing/Webbook2Ed/glossary.htm
R1048	http://www.w3.org/TR/2004/REC-owl-features-20040210/
R1049	http://www.w3.org/TR/2002/NOTE-wscl10-20020314/
R1050	Wikipedia
R1051	DoD Meta Data Registry for XSLT samples. [http://diides.ncr.disa.mil/mdregHomePage/mdregHome.portal]
R1052	XSL Transformations (XSLT) Version 1.0, W3C Recommendation 16 November 1999 [http://www.w3.org/TR/xslt]
R1053	XSLT 2.0 (W3C Working Draft, 5 November 2004) [http://www.w3.org/TR/xslt20]
R1054	XSL Transformations (XSLT) Version 2.0
R1055	XSL (Extensible Stylesheet Language) 1.0, presentation rules to transform a document. [See http://www.w3.org/Style/XSL and http://www.w3.org/TR/xsl]
R1056	The Extensible Stylesheet Language Family http://www.w3.org/Style/XSL
R1058	CSS (Cascading Style Sheets) versions 1 (CSS1) and 2 (CSS2) [See http://www.w3.org/Style/CSS, http://www.w3.org/TR/REC-CSS1, http://www.w3.org/TR/REC-CSS2]
R1070	Multilateral Interoperability Programme (MIP) , **Command, Control and Consultation Information Exchange Data Model (JC3IEDM)** . [http://www.mip-site.org/]
R1071	Various methodologies, such as **refactoring**, support the determination of **interfaces**. **Refactoring** generally applies to the entire software implementation, but is especially helpful in properly flushing out **interfaces**.
R1072	There are other approaches defined through the pattern community: http://hillside.net/patterns/.
R1073	See chapter 3 of Java Design: Building Better Apps &
R1075	For Microsoft Visual Studio 6 and earlier, use the open-source tools described in C/C++ Javadoc guidelines.

R1077	AWT - http://java.sun.com/products/jdk/awt/
R1078	Swing - http://java.sun.com/products/jfc/
R1091	"Web Portal Design Guide 1.1 July 2006," Space and Naval Warfare Systems Center San Diego
R1092	For information on WSRP access specifications, see http://www-106.ibm.com/developerworks/webservices/library/ws-wsrp/
R1093	For information on JSR-168 access specifications, see http://www.jcp.org/aboutJava/communityprocess/final/jsr168/
R1094	For information on portlets, JSR-168, and WSRP access, see http://www.portletsfactory.com/resources/portlets-jsr168-wsrp-portals-books.html
R1095	The Navy Enterprise Application Development Guide (NEADG) provides developer's guidance for the Task Force Web (TFW) Navy Enterprise Portal (NEP).
R1097	The Air Force Portal is built upon the BroadVision portal (http://www.broadvision.com/). For specifics on this portal, refer to http://www.gcss-af.com/.
R1098	Other Air Force portal initiatives are based on the Plumtree portal (http://www.plumtree.com/).
R1101	Adapter pattern - http://c2.com/cgi/wiki?AdapterPattern
R1102	Design patterns: Proxy - http://www.dofactory.com/Patterns/PatternProxy.aspx
R1103	Facade pattern - http://c2.com/cgi/wiki?FacadePattern
R1108	Software Communications Architecture
R1109	Minimum CORBA v1.0
R1110	OMG Lightweight Log Service
R1111	Software-based Communications DTF
R1112	Software Communications Architecture (Wikipedia)
R1116	W3C Extensible Markup Language (XML)
R1117	http://xfront.com/BestPracticesHomepage.html
R1118	Microsoft Standards Reference - http://msdn2.microsoft.com/en-us/library/ms256177.aspx
R1119	Component Organization and Registration Environment - https://www.collab.core.gov/CommunityBrowser.aspx?id=2234
R1120	Federal XML Naming and Design Rules - http://xml.coverpages.org/Federal-NDR-20050609.pdf
R1121	W3C Extensible Markup Language (XML) 1.0 (Fourth Edition) - http://www.w3.org/TR/2006/REC-xml-20060816/

R1122	W3 Schools XML Syntax Rules Tutorial - http://www.w3schools.com/xml/xml_syntax.asp
R1123	W3 Schools XML Validator Tutorial - http://www.w3schools.com/xml/xml_validator.asp
R1124	http://www.w3.org/2001/XMLSchema.xsd
R1125	http://www.xfront.com/xml-schema.html
R1126	http://www.xfront.com
R1127	http://www.xfront.com/ZeroOneOrManyNamespaces.pdf
R1128	http://www.xfront.com/DefaultNamespace.pdf
R1129	Extensible Content Models - http://www.xfront.com/ExtensibleContentModels.pdf
R1130	Element versus Type - http://www.xfront.com/ElementVersusType.pdf
R1131	XML Schema Part 2: Datatypes Second Edition - http://www.w3.org/TR/xmlschema-2/#built-in-datatypes
R1132	Composition versus Subclassing - http://www.xfront.com/composition-versus-subclassing.html
R1133	http://www.w3.org/TR/xpath
R1134	http://www.w3schools.com/xpath/default.asp
R1135	XPath (Wikipedia) - http://en.wikipedia.org/wiki/XPath
R1136	HOWTO: Write Namespace-Agnostic XPath and XSLT - http://jcooney.net/archive/2005/08/09/6517.aspx
R1137	http://www.w3.org/DOM/
R1138	http://www.saxproject.org/
R1140	Human Engineering MIL-STD 1472F, section 5.14 "User Computer Interface"
R1141	"Guide for developing usable and useful web sites" [http://usability.gov]
R1142	"Microsoft Windows User Experience: Official Guidelines for User Interface Developers and Designers," Redmond, WA: Microsoft Press, 1999
R1143	"Apple Human Interface Guidelines," Apple Computer, Inc., 2004 [http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHIGuidelines/index.html]
R1144	"Seven Common Mistakes in Designing a Usable Portal," Iozzo, N., 2002 [http://www.tandemseven.com/pdf/T7_Seven_Common_Mistakes.pdf]
R1145	"The GNOME Usability Project. GNOME Human Interface Guidelines (1.0)," [http://developer.gnome.org/projects/gup/hig/1.0/]
R1146	"Java Look and Feel Design Guidelines: Advanced Topics," Sun Microsystems, Inc., 2001

R1147	"Java Look and Feel Design Guidelines. Second Edition," Sun Microsystems, Inc., 2001
R1148	"Designing Interfaces: Patterns for Effective Interaction Design," Tidwell, J., O'Reilly Media, Inc., 2006
R1149	"Common Presentation Layer Guide Standard 03-01," NAVSEA, September 2006
R1151	"Web-Based Portal Computer-Human Interface Guidelines," Ahlstrom, V. & Allendoerfer, K., 2004 [http://hf.tc.faa.gov/products/bibliographic/tn0423.htm]
R1152	"Web Application Design Handbook: Best Practices for Web-Based Software," Fowler, S. & Stanwick, V., San Francisco: Morgan Kaufmann Publishers, 2004.
R1154	"Federal IT Accessibility Initiative," [http://www.section508.gov/]
R1155	"Electronic and Information Technology Accessibility Standards," Federal Register, [http://www.access-board.gov/sec508/508standards.pdf]
R1156	"Web Content Accessibility Guidelines 1.0," W3C, [http://www.w3.org/TR/WAI-WEBCONTENT/]
R1157	"Guidelines for Keyboard User Interface Design," Microsoft Corporation [http://msdn.microsoft.com/library/?url=/library/en-us/dnacc/html/ATG_KeyboardShortcuts.asp]
R1159	"Internationalization Best Practices: Specifying Language in XHTML & HTML Content," W3C, [http://www.w3.org/TR/i18n-html-tech-lang/]
R1160	"Internationalization Quick Tips for the Web," W3C [http://www.w3.org/International/quicktips/]
R1161	"Developing and Localizing International Software," Madell, T., Parsons, C. & Abegg, J., Englewood Cliffs, NJ: Prentice Hall, 1994
R1162	"Programming for the World: A Guide to Internationalization," O'Donnell, S.M., Englewood Cliffs, NJ: Prentice Hall, 1994
R1163	"Software Internationalization and Localization: An Introduction," Uren, E., Howard, R. & Perinotti,T., New York: Van Nostrand Reinhold, 1993
R1164	DoD Directive 5000.01, *The Defense Acquisition System*, 12 May 2003 (certified current as of 20 November 2007); http://www.dtic.mil/whs/directives/corres/pdf/500001p.pdf.
R1165	DoD Instruction 5000.02, *Operation of the Defense Acquisition System*, 8 December 2008; http://www.dtic.mil/whs/directives/corres/pdf/500002p.pdf.
R1167	DoD Directive 4630.05, *Interoperability and Supportability of Information Technology (IT) and National Security Systems (NSS)*, 05 May 2004 (certified current as of 23 April 2007); http://www.dtic.mil/whs/directives/corres/pdf/463005p.pdf.
R1168	DoD Instruction 4630.8, *Procedures for Interoperability and Supportability of Information Technology (IT) and National Security Systems (NSS)*, 30 June 2004; http://www.dtic.mil/whs/directives/corres/pdf/463008p.pdf.
R1170	*DoD Global Information Grid (GIG) Architecture*, Version 2.0, August 2003.

R1171	DoD Deputy CIO None , **DoD Architecture Framework (DoDAF)** . [http://cio-nii.defense.gov/sites/dodaf20/]
R1172	*DoD Net-Centric Data Strategy*, DoD Chief Information Officer, 9 May 2003, http://www.defenselink.mil/cio-nii/docs/Net-Centric-Data-Strategy-2003-05-092.pdf
R1173	CJCSI 3170.01G, *Joint Capabilities Integration and Development System*, 01 March 2009; http://www.dtic.mil/cjcs_directives/cdata/unlimit/3170_01.pdf.
R1174	CJCSM 3170.01C, *Operation of the Joint Capabilities Integration and Development System*, 01 May 2007; http://www.dtic.mil/cjcs_directives/cdata/unlimit/m317001.pdf.
R1175	CJCSI 6212.01E, *Interoperability and Supportability of Information Technology and National Security Systems*, 15 December 2008; http://www.dtic.mil/cjcs_directives/cdata/unlimit/6212_01.pdf.
R1176	*Net-Centric Operations and Warfare Reference Model (NCOW RM)*, v1.1, 17 November 2005.
R1177	*Net-Centric Checklist*, V2.1.3, Office of the Assistant Secretary of Defense for Networks and Information Integration/Department of Defense Chief Information Officer, 12 May 2004; http://www.defenselink.mil/cio-nii/docs/NetCentric_Checklist_v2-1-3_.pdf.
R1178	*A Modular Open Systems Approach (MOSA) to Acquisition*, Version 2.0, September 2004; http://www.acq.osd.mil/osjtf/mosapart.html.
R1179	*DoD IT Standards Registry* (*DISR*); http://disronline.disa.mil.
R1180	*Net-Centric Attributes List*, Office of the Assistant Secretary of Defense for Networks and Information Integration/Department of Defense Chief Information Officer, 2 February 2007; http://www.defenselink.mil/cio-nii/docs/NetCentricAttributesOfficial.pdf .
R1181	*Global Information Grid (GIG) Key Interface Profiles (KIPs) Framework* (DRAFT), Version 0.95, 7 October 2005.
R1182	Office of the Under Secretary of Defense (USD) for Acquisition, Technology and Logistics (AT&L) memorandum, *Instructions for Modular Open Systems Approach (MOSA) Implementation*, 7 July 2004, available at www.acq.osd.mil/osjtf
R1184	Program Executive Office, Integrated Warfare Systems (PEO-IWS 7), *Naval Open Architecture Contract Guidebook*, Version 1.1, 25 October 2007 available via the Defense Acquisition University Acquisition Community Connection Web site (https://acc.dau.mil/oa) by following the "Policy & Guidance" link.
R1185	GAO Report to Congressional Committees, Weapons Acquisition, *DOD Should Strengthen Polices for Assessing Technical Data Needs to Support Weapon Systems*, GAO-06-839, July 2006
R1189	For *Open Architecture Assessment Tool (OAAT)* information access the Defense Acquisition University (DAU) Web site located at https://acc.dau.mil/CommunityBrowser.aspx?id=18016
R1190	DoD CIO memos: • 9 June 2003, *Internet Protocol Version 6 (IPv6)* • 29 September 2003, *Internet Protocol Version 6 (IPv6) Interim Transition Guidance*

	• 28 November 2003, *Internet Protocol Version 6 (IPv6) Transition Plan Coordination and Interim Tasking* • 16 August 2005, *Internet Protocol Version 6 (Ipv6) Policy Update* • 16 August 2005, *DoD Internet Protocol Version 6 (IPv6) Pilot Nominations*
R1191	DoD Directive O-8530.1, *Computer Network Defense*
R1192	DoD Instruction O-8530.2, *Support to Computer Network Defense Services (CNDS)*
R1193	See the following items from the Defense Acquisition Guidebook: • Compliance with the Net-Centric Operations and Warfare Reference Model • Compliance with Applicable Global Information Grid Key Interface Profiles • Compliance with DoD Information Assurance Requirements • Supporting Integrated Prchitecture Products
R1194	DoD Directive 5000.01, Enclosure 1, Paragraph E1.9, Information Assurance Acquisition managers shall address information assurance requirements for all weapon systems; Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance systems; and information technology programs that depend on external information sources or provide information to other DoD systems. DoD policy for information assurance of information technology, including NSS, appears in DoD Directive 8500.01.
R1197	DoD Directive 8500.01E, *Information Assurance (IA)*, 24 October 2004 (certified current as of 23 April 2007). This directive establishes policy and assigns responsibilities under 10 U.S.C. 2224 to achieve Department of Defense information assurance (IA) through a defense-in-depth approach that integrates the capabilities of personnel, operations, and technology, and supports the evolution to net-centric warfare.
R1198	DoD Instruction 8500.2, *Information Assurance (IA) Implementation* ***This instruction implements policy, assigns responsibilities, and prescribes procedures for applying integrated, layered protection of the DoD information systems and networks under DoD Directive 8500.1.***[R1197]
R1199	DoD Instruction 8580.1, *Information Assurance (IA) in the Defense Acquisition System* ***This instruction implements policy, assigns responsibilities, and prescribes procedures necessary to integrate Information Assurance (IA) into the Defense Acquisition System; describes required and recommended levels of IA activities relative to the acquisition of systems and services; describes the essential elements of an Acquisition IA Strategy, its applicability, and prescribes an Acquisition IA Strategy submission and review process.***
R1202	**OMG** Data Distribution Service for Real-time Systems Version 1.2
R1203	**OMG** Data Distribution Portal (http://portals.omg.org/dds)
R1204	24 June 2005, *Air Force Internet Protocol Version 6 (IPv6) Policy and Transition Plan Tasking*
R1205	June 2006, *DoD IPv6 Transition Plan*, Version 2.0
R1206	DoD Instruction 8520.2; 1 April 2004; *Public Key Infrastructure (PKI) and Public Key (PK) Enabling*; http://www.dtic.mil/whs/directives/corres/pdf/852002p.pdf

R1207	David Sprott *"Service Oriented Architecture: An Introduction for Managers"*; July 2004, http://www.ibm.com/services/us/bcs/pdf/soa-cbdi-report-2004-july.pdf
R1211	AFEI NCOIF report *"Industry Best Practices in Achieving Service Oriented Architecture (SOA)"*; April 22, 2005; http://www.afei.org/news/documents/IndustryBestPracticesforAchievingSOA_000.pdf
R1215	Yefim V. Natis, Gartner; *"Service-Oriented Architecture Scenario"*; 16 April 2003; http://www.gartner.com/DisplayDocument?doc_cd=114358
R1217	DoD 8320.02-G, 12 April 2006, *Guidance for Implementing Net-Centric Data Sharing*; http://www.dtic.mil/whs/directives/corres/pdf/832002g.pdf
R1218	AF ESC Net-Centric Data Strategy Implementation Roadmap, Chief Architect's Office, 5/23/2003, Draft v 0.83
R1219	Dr. Mark Kramer, *"Towards Implementation of the DoD Net-Centric Data Strategy (NCDS)"*; May 2007 (presentation)
R1220	Yvonne Perlmutter, *"653 ELSG Cross Portfolio Acquisitions Assessment"*; 17 July 2007 (presentation)
R1222	NIST SP 800-95, "Guide to Secure Web Services" dated August 2007 http://csrc.nist.gov/publications/nistpubs/800-95/SP800-95.pdf
R1223	Web Services Interoperability (WS-I) Profiles: http://www.ws-i.org/deliverables/Default.aspx
R1224	DoD Chief Information Officer Memorandum, ***DoD Net-Centric Data Management Strategy - Metadata Registration***, 3 April 2003 (available at http://www.defenselink.mil/cio-nii/docs/DMmemo20030403.pdf)
R1225	DoD Discovery Metadata Specification (DDMS); refer to the DDMS homepage for current version information: http://metadata.dod.mil/mdr/irs/DDMS/
R1227	DoD Metadata Registry and Clearinghouse, http://xml.dod.mil
R1228	ISO/IEC Standard 11179, ***Information Technology # Metadata Registries (MDR)***, Parts 1-6, available from the International Organization for Standardization (ISO), http://www.iso.org
R1229	XML Schema Specification, World Wide Web Consortium (W3C), http://www.w3c.org/XML
R1232	DoD Directive 5230.09, *Clearance of DoD Information for Public Release*, 22 August 2008
R1235	CJCSM 3170.01B, Operation of the Joint Capabilities Integration and Development System, 11 May 2005
R1237	Web Services Interoperability (WS-I) Basic Security Profile, http://www.ws-i-org
R1239	NCIDs Global Information Grid Net-Centric Iimplementation Document - Service Definition Framework (S300), 21 December 2005
R1240	ASD(NII)/DoDCIO Memo, Subject: Radio Frequency (RF) Equipment Acquisition Policy (JTRS), 17 June 2003

R1241	http://jtrs.army.mil (SCA)
R1244	DISA ***Information Assurance Support Environment*** Web site, http://iase.disa.mil
R1245	DoD Directive 8320.02, ***Data Sharing in a Net-Centric Department of Defense***, December 2, 2004 (certified current as of 23 April 2007)
R1246	OASIS , **Web Services Security: SAML Token Profile 1 .1** . [http://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf]
R1247	Director of Central Intelligence Directive 6/3, ***Protecting Sensitive Compartmented Information within Information Systems***, 5 June 1999
R1249	Air Force Instruction 33-202, "Network and Computing Security", 26 September 2003
R1251	DoD CIO Guidance and Policy Memorandum 6-8510, "DoD GIG Information Assurance", 16 June 2000.
R1252	DoD End-to-End Information Assurance of GIG, Version 1.0, 30 June 2004
R1255	Green, David and Grillo, Bob, SRI International, "The State of IPv6 - A DoD Prospective, February 2005 (prepared for the DoD IPv6 Standards Working Group
R1256	International Organization for Standardization (ISO) Open Systems Interconnection Basic Reference Model (OSI Model)
R1257	Blake, S., Black D., Carlson, M., Davies, E., Wang, Z. and W. Weiss, ***An Architecture for Differentiated Services***, RFC 2475, IETF, December 1998.
R1258	Assistant Secretary of Defense for Networks and Information Integration, Memorandum; *Joint Net-Centric Capabilities*, 15 July 2003
R1259	Defense Information Systems Agency, Net-Centric Enterprise Services (NCES) Program Management Office, http://www.disa.mil/nces/index.html
R1262	The TeleManagement Forum's Enhanced Telecom Operations MapTM (eTOM) and the Information Technology Infrastructure Library (ITILR)
R1276	XML Namespaces, Version 1.0, http://www.w3.org/TR/REC-xml-names, Version 1.1 http://www.w3.org/TR/xml-names11
R1280	Universal Description, Discovery, and Integration (**UDDI**), http://www.oasis-open.org/committees/uddi-spec
R1283	Net-Centric Environment Joint Functional Concept, Version 1.0, April 7, 2005
R1284	Net-Centric Operational Environment Joint Integrating Concept, Version .08, August 26, 2005
R1288	Deputy Under Secretary of Defense for Advanced Systems and Concepts, ***Open Technology Development Roadmap Plan***, April 2006; http://www.acq.osd.mil/jctd/articles/OTDRoadmapFinal.pdf
R1289	Javadoc Tool Home Page, http://java.sun.com/j2se/javadoc/

R1290	XML Documentation Comments (C# Programming Guide), http://msdn2.microsoft.com/en-us/library/b2s063f7.aspx
R1291	DoD Instruction 8510.01, DoD Information Assurance Certification and Accreditation Process (DIACAP), 28 November 2007; available at http://www.dtic.mil/whs/directives/corres/pdf/851001p.pdf (superseded DoD Instruction 5200.40, DITSCAP)
R1292	DoD Instruction 8552.01, ***Use of Mobile Code Technologies in DoD Information Systems***, 23 October 2006 (available at http://www.dtic.mil/whs/directives/corres/pdf/855201p.pdf)
R1293	Space and Naval Warfare Systems Command (SPAWAR), ***DOD Domain Controller Public Key Infrastructure (DOD-PKI) Domain Controller Administrator Operations Guide (DCAOP)***, 30 May 2006; https://infosec.navy.mil/clt/index.jsp (user registration and DoD PKI Certificate required for access)
R1294	United States Air Force Public Key Infrastructure System Program Office (USAF PKI SPO), ***Configuration and Operations Guide For Air Force Smart Card Certificate-Based Logon Using DoD PKI Domain Controller Certificates***, April 2006; https://afpki.lackland.af.mil/html/sclogon.asp (DoD PKI Certificate required for access)
R1295	Army IA NETCOM, ***Common Access Card (CAC) Cryptographic Logon (CCL) Technical Configuration Guide***, V 1.0, February 2006; https://www.us.army.mil/suite/page/237211; user account (Army or Defense Knowledge Online, AKO or DKO) and DoD PKI Certificate required for access)
R1296	USMC, ***Cryptographic LogOn Enabler (CLOE) version 1 formerly Logon EDI-PI Attribute Populator (LEAP) Installation, Configuration and Operations Guide***, February 2006. https://www2.mcnosc.usmc.mil/NR/rdonlyres/38542F70-263A-469C-B5E8-7F2002D85EF6/0/CLOE_Operations_Guide.doc (DoD PKI Certificate required for access)
R1297	DoD Directive 8190.3, ***Smart Card Technology***, 31 August 2002; http://www.dtic.mil/whs/directives/corres/pdf/819003p.pdf
R1298	Carnegie Mellon University Software Engineering Institute CERT, ***Secure Coding Standards***; https://www.securecoding.cert.org/confluence/display/seccode/CERT+Secure+Coding+Standards
R1299	Common Weakness Enumeration; http://cwe.mitre.org/index.html
R1300	Open Web Application Security Project (OWASP), ***Top Ten Most Critical Web Application Security Vulnerabilities***; http://www.owasp.org/index.php/OWASP_Top_Ten_Project
R1301	Carnegie Mellon University Software Engineering Institute CERT, ***C++ Secure Coding Standard***; https://www.securecoding.cert.org/confluence/display/seccode/CERT+Secure+Coding+Standards
R1302	Microsoft Developer Network (MSDN), ***Secure Coding Guidelines for the .NET Framework***; http://msdn2.microsoft.com/en-us/library/aa302372.aspx
R1303	Sun Microsystems, ***Secure Coding Guidelines for the Java Programming Language***; http://java.sun.com/security/seccodeguide.html
R1304	University of Virginia, Department of Computer Science, Inexpensive Program Analysis Group, ***Splint - Secure Programming Lint***; http://lclint.cs.virginia.edu/

R1305	Sun Microsystems, ***Java Annotations***; http://java.sun.com/docs/books/tutorial/java/javaOO/annotations.html
R1306	Microsoft Developer Network (MSDN), ***Selective Nodification of the Behavior of Compiler Warning Messages***; http://msdn2.microsoft.com/en-us/library/ms879818.aspx
R1307	IBM, ***Open Architecture Principles and Guidelines***, v1.5.4, 19 September 2007; available via https://acc.dau.mil/CommunityBrowser.aspx?id=170302
R1308	OAISIS None , **Reference Architecture for Service Oriented Architecture Version 1.0 Public Review Draft 1** . [http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-pr-01.pdf]
R1312	*DoD Net-Centric Data Strategy*, DoD Chief Information Officer, 9 May 2003; http://www.defenselink.mil/cio-nii/docs/Net-Centric-Data-Strategy-2003-05-092.pdf
R1313	*DoD Net-Centric Services Strategy*, DoD CIO, 4 May 2007, http://www.defenselink.mil/cio-nii/docs/Services_Strategy.pdf
R1314	Internet Corporation for Assigned Names and Numbers, http://www.icann.org/
R1315	Load balancing (computing), http://en.wikipedia.org/wiki/Load_balancing_(computing)
R1316	Blue Coat *Web Applications* (Optimization Content partial source http://www.bluecoat.com/solutions/enterprise/controlperformance/webapplications)
R1317	NIST, *Cryptographic Algorithms and Key Sizes for Personal Identity Verification*, http://csrc.nist.gov/publications/nistpubs/800-78-1/SP-800-78-1_final2.pdf
R1318	NIST, *A Comparison of the Security Requirements for Crytographics Modules in FIPS 140-1 and FIPS 140-2*, http://csrc.nist.gov/publications/nistpubs/800-29/sp800-29.pdf
R1319	*Dynamic Host Configuration Protocol*, RFC 2131, http://tools.ietf.org/html/rfc2131, March 1997, Internet Engineering Task Force (IETF) Network Working Group
R1320	*Domain Names - Implementation and Specification*, RFC 1035, http://tools.ietf.org/html/rfc1035, November 1987, Internet Engineering Task Force (IETF) Network Working Group
R1321	*DNS Extensions to Support IP Version 6*, RFC 3596, http://tools.ietf.org/html/rfc3596, October 2003, Internet Engineering Task Force (IETF) Network Working Group
R1322	*Network Time Protocol (Version 3) Specification, Implementation and Analysis*, RFC 1305, http:// www.ietf.org/rfc/rfc1305.txt, March 1992, Internet Engineering Task Force (IETF) Network Working Group
R1323	*Internet Time Synchronization: The Network Time Protocol*, RFC 1129, http://www.ietf.org/rfc/rfc1129.pdf, October 1989, Internet Engineering Task Force (IETF) Network Working Group
R1324	*Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI*, RFC 4330, http://tools.ietf.org/rfc/rfc4330.txt, January 2006, Internet Engineering Task Force (IETF) Network Working Group
R1325	Dynamic Updates in the Domain Name System (DNS UPDATE), RFC 2136, http://tools.ietf.org/html/rfc2136, April 1997, Internet Engineering Task Force (IETF) Network Working Group

R1326	*The DHCP Handbook*, ISBN: 1-57870-137-6, 1999, Ralph Droms, Ted Lemon, The Mcmillan Technical Publishing, Indianpolis, IN, USA
R1327	*DHCP Options and BOOTP Vendor Extensions*, RFC 2132, http://tools.ietf.org/html/rfc2132, March 1997
R1328	*DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*, http://tools.ietf.org/html/rfc3646, RFC 3646, December 2003
R1329	DoD None , **Ports, Protocols, and Services Management (PPSM)** DISA . [http://iase.disa.mil/ports/index.html]
R1330	DoD Memorandum , **Encryption of Sensitive Unclassified Data at Rest on Mobile Computing Devices and Removable Storage Media** Chief Information Officer . [http://iase.disa.mil/policy-guidance/dod-dar-tpm-decree07-03-07.pdf]
R1331	DoD Directive 8100.02, **Use of Commercial Wireless Devices, Services, and Technologies in the Department of Defense (DoD) Global Information Grid (GIG)** . [http://www.dtic.mil/whs/directives/corres/pdf/810002p.pdf]
R1332	DoD Memorandum , **Department of Defense Guidance on Protecting Personally Identifiable Information (PII)** . [http://iase.disa.mil/policy-guidance/pii-signed-memo-08182006.pdf]
R1333	DoD Memorandum , **Protection of Sensitive DoD Data at Rest on Portable Computing Devices** .
R1334	General Services Administration (GSA) Memo 10359, **Data at Rest (DAR) Encryption Awardees Announced** . [http://www.gsa.gov/Portal/gsa/ep/contentView.do?contentType=GSA_BASIC&contentId=23172&noc=T]
R1335	DoD Deputy CIO None , **Defense Enterprise Information Architecture** . [http://www.defenselink.mil/cio-nii/sites/diea/overview.html]
R1336	The Internet Society , **The Transition to IPv6** Internet Society (ISOC) . [http://www.isoc.org/briefings/006/isocbriefing06.pdf]
R1337	Internet Engineering Task Force (IETF) Network Working Group , **Terminology for Policy-Based Management RFC3198** . [http://tools.ietf.org/html/rfc3198]
R1338	Information Assurance Technology Analysis Center (IATIC) , **Software Security Assurance: A State-of-the-Art Report (SOAR)** . [http://iac.dtic.mil/iatac/download/security.pdf]
R1339	Committee on National Security Systems (CNSS) Instruction 4009, **National Information Assurance (IA) Glossary** . [http://www.cnss.gov/Assets/pdf/cnssi_4009.pdf]
R1340	Software Assurance (SwA) Acquisition Working Group None 8510.01, **Software Assurance in Acquisition: Mitigating Risks to the Enterprise** ASD(NII)/CIO . [https://buildsecurityin.us-cert.gov/swa/downloads/SwA_in_Acquisition_102208.pdf]
R1341	DoD Directive 8000.01, **Management of the Department of Defense Information Enterprise** ASD(NII)/DoD CIO . [http://www.dtic.mil/whs/directives/corres/pdf/800001p.pdf]
R1342	DoD CIO None , **Department of Defense Global Information Grid Architectural Vision** . [http://cio-nii.defense.gov/docs/GIGArchVision.pdf]

R1343	DoD None , **Net-Centric Environment Joint Functional Concept** . [http://www.dtic.mil/futurejointwarfare/concepts/netcentric_jfc.pdf]
R1344	DoD Directive 5144.1, **Assistant Secretary of Defense for Networks and Information Integration/DoD Chief Information Officer (ASD(NII)/DoD CIO)** . [http://www.dtic.mil/whs/directives/corres/pdf/514401p.pdf]
R1345	ASD(NII)/DoD CIO None , **Deputy Assistant Secretary of Defense for Cyber, Identity, and Information Assurance Strategy** . [http://cio-nii.defense.gov/docs/DoD_IA_Strategic_Plan.pdf]