Net-Centric Implementation Framework

Part 1: Overview

Part 2: ASD(NII) Checklist Guidance

Part 3: Migration Guidance

Part 4: Node Guidance

Part 5: Developer Guidance

Part 6: Contracting Guidance for Acquisition

V 2.1.0

12 October 2007

Net-Centric Enterprise Solutions for Interoperability (NESI) is a collaborative activity of the USN Program Executive Office for Command, Control, Communications, Computers and Intelligence (PEO C4I); the USAF Electronic Systems Center (ESC); and the Defense Information Systems Agency (DISA).

NESI-X Subdocument generated using **View, NESI Part 3: Migration Guidance**
Generated: **Wed Oct 31 18:13:51 PDT 2007**
NESI-X Version: **v1.0.0 build 1819 - 2007/08/23 15:26**

Table of Contents

Perspectives

P1117: NESI Overview

Net-Centric Enterprise Solutions for Interoperability (***NESI***) provides, for all phases of the acquisition of net-centric solutions, actionable guidance that meets DoD Network-Centric Warfare goals. The guidance in NESI is derived from the higher level, more abstract concepts provided in various directives, policies and mandates such as the Net-Centric Operations and Warfare Reference Model (NCOW RM) [R1176] and the ASD(NII) Net-Centric Checklist [R1177] . As currently structured, NESI implementation covers architecture, design and implementation; compliance checklists; and a collaboration environment that includes a repository.

More specifically, NESI is a body of architectural and engineering knowledge that guides the design, implementation, maintenance, evolution, and use of the Information Technology (IT) portion of net-centric solutions for military application. NESI provides specific technical recommendations that a DoD organization can use as references. Stated another way, NESI serves as a reference set of compliant instantiations of these directives.

NESI is derived from a studied examination of enterprise-level needs and, more importantly, from the collective practical experience of recent and on-going program-level implementations. It is based on today's technologies and probable near-term technology developments. It describes the practical experience of system developers within the context of a minimal top-down technical framework. Most, if not all, of the guidance in NESI is in line with commercial best practices in the area of enterprise computing.

NESI applies to all phases of the acquisition process as defined in DoD Directive 5000.1 [R1164] and DoD Instruction 5000.2 [R1165] and to both new and legacy programs. NESI provides explicit counsel for building in net-centricity from the ground up 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.

Content Structure

Perspective	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 amplifying 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

This document has been cleared for public release by competent authority in accordance with DoD Directive 5230.9 and is granted Distribution Statement A: Approved for public release; distribution is unlimited. Obtain electronic copies of this document at http://nesipublic.spawar.navy.mil.

Vendor Neutrality

The 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 whatever tools the contributor was using or knew best. However, the products described 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 open-source tools that appear as examples in this guide. Similarly, any lists of products or vendors are intended only as references or starting points, and 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-source 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.

Collaboration Site

The Navy has established a collaboration site to support NESI community interaction. It is located at https://nesi.spawar.navy.mil (user registration required). Use this site for collaborative software development across distributed teams.

References

- R1164: DoD Directive 5000.1, *The Defense Acquisition System*, 12 May 2003 (certified current as of 24 November 2003); http://www.dtic.mil/whs/directives/corres/pdf/500001p.pdf.
- R1165: DoD Instruction 5000.2, *Operation of the Defense Acquisition System*, 12 May 2003; http://www.dtic.mil/whs/directives/corres/pdf/500002p.pdf.
- R1166: DoD Directive 8100.1, *Global Information Grid (GIG) Overarching Policy*, 19 September 2002 (certified current as of 21 November 2003); http://www.dtic.mil/whs/directives/corres/pdf/810001p.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 Architecture Framework (DoDAF)*, Version 1.5, 23 April 2007; https://dars1.army.mil/IER/index.jsp
- 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 3170.01F, *Joint Capabilities Integration and Development System*, 01 May 2007; http://www.dtic.mil/cjcs_directives/cdata/unlimit/3170_01new.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 6212.01D, *Interoperability and Supportability of Information Technology and National Security Systems*, 8 March 2006; 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.

P1198: NESI Part 3: Migration Guidance

NESI Part 3: Migration Guidance is the third of six parts of the NESI Implementation documentation set. The intended audience for this content is government and industry program managers and system/software engineers of existing programs. It presents an approach for migrating deployed applications to greater degrees of net-centricity and interoperability. It describes the implementation of a phased software migration strategy for delivering net-centric capability while fulfilling current contractual and program maintenance obligations.

This material is a significant revision of NESI Part 3 v2.0 of 30 April 2007. It introduces an architecture-based approach that identifies explicit consideration for migration to a Service-Oriented Architecture (SOA). While continuing to emphasize an incremental approach, this revision removes the explicit mention of NESI Migration Levels found in NESI Part 3 v2.0. Instead, this revision provides a set of more flexible migration patterns organized by approximate migration starting points that are similar in nature to the NESI Migration Levels found in NESI v2.0. This revision also includes a discussion of factors to consider during migration and a detailed discussion about the process of migration. This revision does not explicitly address migration from the Common Operating Environment; however, NESI Part 3 v2.0 does provide information concerning migrating COE systems and applications and is available on the NESI Public Site, http://nesipublic.spawar.navy.mil/.

The material in this revieion is based on industry best practices and is designed to evolve by absorbing "lessons learned" by DoD enterprise-level and program personnel as they gain experience with net-centricity and SOA. This revision emphasizes using a business case analysis to determine the specific approach to migration to net-centricity and SOA.

Detailed Perspectives

- Migration Introduction
- Migration Planning Process
- Migration Patterns
- Critical Migration Concerns
- Net-Centric and SOA Assessments
- Net-Centric Data Strategy (NCDS)

P1199: Migration Introduction

NESI Part 3: Migration Planning presents a methodology that a Program Management Office can use when planning and executing the migration of legacy systems to net-centricity. It is derived from industry best practices and guides the reader through a sequence of decision making steps leading to the development of an executable migration plan. It presents a phased, iterative approach emphasizing warfighter benefit as the end migration goal. It is oriented to the use of Service-Oriented Architecture (SOA) concepts as a primary migration approach.

Most acquisition programs present a complex management challenge and generally require unique migration paths. NESI Part 3 presents five high-level activities that can serve as the foundation for the development of a program-unique migration plan. In addition, it contains examples of migration decisions and their outcomes based on the initial net-centric state of a system.

Migration Guidance Purpose and Scope

NESI Part 3 helps DoD programs develop and execute net-centric migration plans using a methodology that is largely based on the concept of net-centric SOA (see SOA and Net-Centricity below). However, this guide also helps programs migrate to net-centricity even when there are no plans to implement services. It addresses key concerns associated with other aspects of net-centricity (e.g., connect via Internet Protocol (IP), share data, protect data). The focus is on the ***process*** of migration. It includes a discussion of a migration plan development methodology that involves technical solution trade-offs.

This guidance addresses fielded systems that are subject to net-centric migration through maintenance as well as minor or major upgrades. Systems that are known to be phased out in the near future are generally not subject to migration. This guide may also be useful for new starts, especially where the newly acquired materiel has to coexist with legacy materiel. While the primary focus is on formal DoD programs, this guidance provides significant background information that may be useful to informal projects such as research and other investigative initiatives.

Migration Guidance Audience

The intended audience for ***NESI Part 3: Migration Guidance*** is the set of stakeholders associated with developing incremental net-centric and interoperable improvements to existing systems. The primary audience within this set is government and industry program managers and system/software engineers. Portions of Part 3 also pertain to other stakeholders such as contracting personnel and end users. This document may also be useful for stakeholders associated with new programs.

Migration Guidance Overview

NESI Part 3 contains the following key perspectives:

- Migration Planning Process discusses a migration planning methodology. It provides a list of actions leading to the development of a multi-phased net-centric migration plan. It uses an architecture-based approach that is centered on the analysis of operational processes (sometimes known as "mission threads") within a service-oriented architecture paradigm. The architecture is represented in time-phased architecture products which represent operational, system, and technical views of a particular context such as an enterprise or a program. An architecture-based approach to migration planning bases migration efforts on the steps required to transition from an "as-is" to a "to-be" architecture. The architecture includes both materiel and non-materiel aspects (such as people and processes). The approach in Part 3 focuses on the following:
    - the assessment of the net-centric and SOA maturity of the as-is and one or more alternative target (to-be) architectures
    - the consideration of multiple concerns that are critical for migration success
    - the development of a business-case analysis for the alternative target architecture(s)

- Migration Patterns discusses patterns that describe typical migration scenarios (starting points, steps to take to migrate to enhanced levels of net-centricity or SOA, and the resulting end points). The starting points present various current levels of net-centricity and infrastructure investment. The description of the steps include considerations of some of the trade-offs associated with them. The end points identify future migration options that could be taken. The patterns help program personnel perform the business case analysis for the program's migration. While multiple patterns may apply to a specific program, the patterns provide an aid for determining potential to-be architectures (both near- and far-term) as well as specific transformation steps for achieving those architectures.

- Critical Migration Concerns contains a high-level explanation of areas that require particular attention during migration to net-centric SOA, addressing each of these concerns in the program's migration plan.

- Net-Centric and SOA Assessment contains material on net-centric assessment tools. Use the guidance in this section during the preparation, planning, and execution stages of the migration to assess the initial state of the systems, project the phased outcomes, measure the actual progress, and update the plans. It is also useful for reporting the results of the migration.

- Net-Centric Data Strategy (NCDS) provides more in-depth guidance on implementing the Net-Centric Data Strategy according to DoD Directive 8320.02, *Data Sharing in a Net-Centric Department of Defense*. For many programs, a large part of the migration to net-centricity and SOA may involve the migration to increased data sharing.

Migration Business Case Analysis

The purpose and true measure of success for any net-centric migration is improved **capability** delivered to the warfighter. Either this improvement occurs directly as a result of functional improvements associated with the migration or it is realized indirectly through non-functional enhancements to the architecture (e.g., increased agility, increased maintainability, increased securability).

Support migration by employing a business case that clearly identifies the benefit in terms of increased warfighter capability at every step of the migration process. The business case includes a consideration of the constraints of cost, risk, and schedule.

SOA and Net-Centricity

The SOA-based approach is well-suited to realizing improved warfighter capability in many circumstances. SOA facilitates integration of legacy assets and potentially reduces development costs by enabling the reuse of SOA-based components. Due to its focus on supporting business processes, it also fosters cross-enterprise collaboration and forces the engineering and information technology community to think in terms of operational processes, thus helping align the technical goals of net-centricity with warfighter needs. In addition, even if some of the integration solutions may not end up being service-based, the overall SOA-based approach to migration enhances operational effectiveness through the focus on improving operational processes.

SOA can be a catalyst for enterprise-integration and net-centricity. There are a number of desirable properties of SOA that are inherently net-centric:

- packaging data in standardized formats

- loosely coupling data and processing that data

- replacing, expanding and/or reusing functionality flexibly

However, since services (the central element of SOA) are only one aspect of net-centricity, it is possible to reach significant levels of SOA maturity and still not be fully net-centric. See the Wrapping Legacy Code into a Service pattern as an example.

Migration Assumptions

This migration guidance is based on the following assumptions:

- Every migration implementation is unique
- Most migrations will require a phased, iterative implementation
- Migration implementations will be a mix of top-down and bottom-up efforts

Each Program Management Office faces a number of migration options and circumstances that determine the uniqueness of the migration path for a particular program. There is no "one size fits all." However, there are some general architectural patterns or templates that describe the approximate start and end points of typical migrations and the recommended migration steps associated with these patterns (see Migration Patterns). A program may be able to leverage one or more of these patterns to the extent that its situation matches that of the known patterns.

"Taking an iterative approach to SOA is a fundamental best practice." [R1212] Because most large-scale SOA migrations are expected to be lengthy, implement the migration in phases. Start with building a realistic phased migration plan. Update the plan at the end of each transition phase to assess the results and to accommodate changes in the environment and any "lessons learned."

While a full-scale SOA migration can be lengthy, there is often a great value in "tactical SOA" (see the ***Early Learning*** and ***Maturity*** phases in the Phases of SOA Adoption perspective). Even early stages of SOA migration, if thoughtfully implemented, can provide a quick improvement in capabilities and valuable lessons learned, both good and bad, for the program and for the enterprise community. Even in spite of the necessity to reengineer services in the future (see the Reengineering phase in the Phases of SOA Adoption perspective), an early services-based integration experience could prove cost-effective.

The migration to SOA is a confluence of top-down, bottom-up, and middle-out efforts. Practice shows that adoption of SOA at the enterprise level is an evolutionary process characterized by building on small, localized "wins" combined with the incremental transformation of the overall business and the gradual development of centralized SOA migration governance. A goal of this guide is to help programs deal with the challenge of implementing their (bottom-up) strategies while working with enterprise integration partners (middle-out) and moving towards becoming part of the whole (top-down), even while the top-down direction is still under development. Most of the concerns and activities listed in the Critical Migration Concerns perspective are directed at this goal.

References

- 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
- R1208: CBDI 2003 report *"Web Services Roadmap"*; http://roadmap.cbdiforum.com
- R1209: AFEI report *"Facilitating Shared-Services in the DoD"*; Feb 12, 2006; http://www.afei.org/news/documents/DS3SharedServicesPaperFinalVersion021206c.pdf
- R1210: Randy Heffner, Forrester Research; Executive Summary from *"Your Path to SOA"*; December 7, 2004; http://www.forrester.com/go?docid=34813
- 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
- R1212: Jason Bloomberg & Ron Schmelzer, ZapThink, LLC *"SOA Pilots"* (presentation as part of a full-day seminar for Tarantell in Oslo, Norway on November 16, 2006); http://www.zapthink.com/report.html?id=ZTP-0277 [user access required]
- R1213: David Margulius, *"Banking on SOA"*; InfoWorld; July 13, 2006; http://www.infoworld.com/article/06/07/13/29FEwachovia_1.html
- R1214: H.M. Sneed, *"Wrapping Legacy Software for Reuse in SOA"*; Dec 2005
- R1215: Yefim V. Natis, Gartner; *"Service-Oriented Architecture Scenario"*; 16 April 2003; http://www.gartner.com/DisplayDocument?doc_cd=114358
- R1216: Global Information Grid Net-Centric Implementation Document - Service Definition Framework (S300); 21 December 2005 (V2.0)
- R1217: DoD 8320.02-G, April 12, 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)

P1200: Migration Planning Process

Most migration efforts, because of their complexity, require a formal migration plan. The migration plan helps to synchronize technology improvements, overall mission/business capability improvements, and the changes in the business practices enabled by SOA while making explicit the return on investment (ROI). The migration plan, while more strategic in nature, serves as the basis for detailed project plans for specific near term migration increments.

In general terms, the migration plan addresses the three interrogatives: why, what, and how. The migration to net-centricity and SOA is driven by the following:

- a vision for the future state of the system within the larger context of the enterprise
- a combination of specific program requirements and enabling technologies
- an assessment of the as-is state of the system (i.e., the as-is architecture)

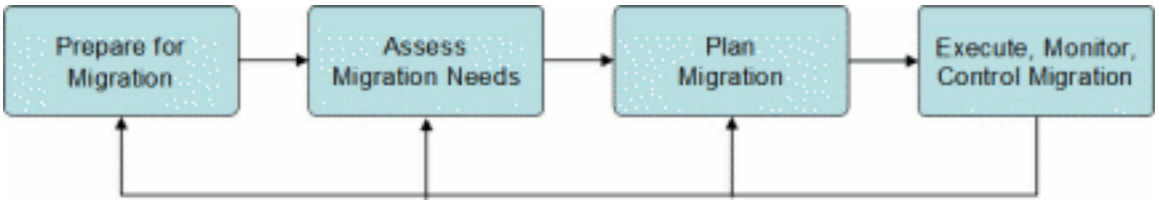
The vision provides a somewhat idealized target for the system within an unspecified timeframe. The requirements and enabling technologies provide external motivation for change within the current timeframe. The assessment of the as-is architecture provides an internal motivation for change within the current timeframe.

There are many ways to depict migration planning. The figure below shows a basic planning and execution loop adapted to net-centric and SOA migration. While the steps are depicted as sequential, the process is in fact iterative and interactive with many opportunities for activities to occur in parallel or otherwise overlap in time. They may not occur in the strict order outlined in this high-level depiction and there is often significant parallelism and reordering of lower-level activities. All of the steps are interrelated and have numerous feedback paths. A key feedback path is from the execution of migration increments into the evolution of the overall migration plan as well as the development of the detailed project plan for the next migration increment in a spiral fashion.

The process can also vary in the breadth and depth to which various sub-steps are executed. For example, documenting elements of a to-be architecture might occur before the as-is architecture documentation is complete.

While not all programs will require significantly detailed documentation associated with the migration, executing the process to at least a cursory degree can ensure that program personnel do not overlook key concerns. At the one extreme, the migration plan can be a simple document indicating the summary results of executing the process. At the other extreme, the migration plan can be a very detailed set of documents that are used to direct a complex set of interrelated tasks.

The migration planning process consists of four main activities as shown below.



Prepare for Migration	compile relevant background information, document the as-is architecture
Assess Migration Needs	assess existing requirements and as-is architecture, develop migration rationale
Plan Migration	develop to-be architecture(s), plan an incremental implementation
Execute, Monitor, and Control Migration	execute program plans, monitor progress, maintain architecture, adjust plans, etc.

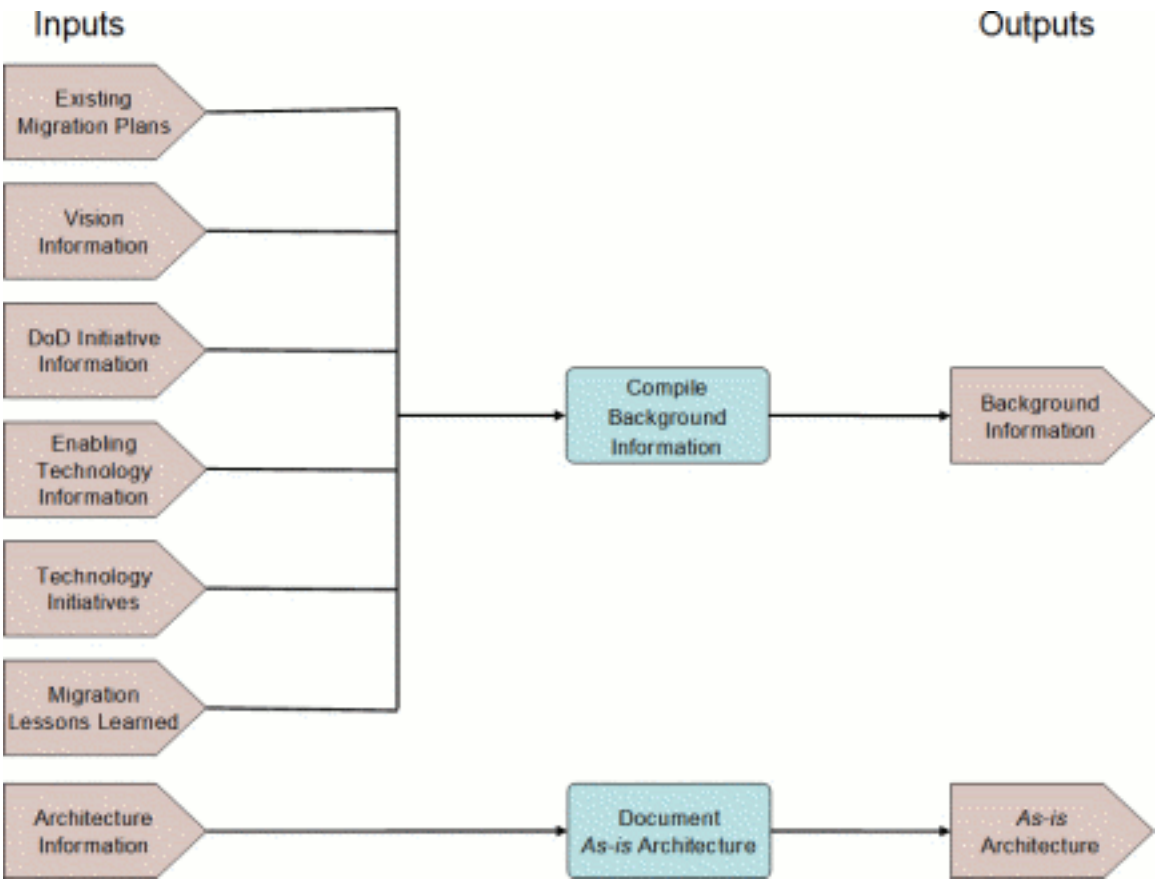
Best Practices

- BP1835: Develop a formal migration plan to support the migration to **net-centricity** and SOA.
- BP1836: Obtain consensus on the migration plan from all key **stakeholders**.
- BP1837: Update the **net-centric** and SOA migration plan in an iterative manner as the program gains migration experience and conditions change.

P1205: Prepare for Migration

To prepare for migration to net-centricity, perform the following actions (as depicted in the diagram below):

- Compile migration background information
- Document the as-is architecture



Compile Migration Background Information

The program starts (or iterates) the migration planning process by compiling (or updating) relevant background information. This includes but is not limited to the following:

- Existing migration documentation
- Higher headquarters documents containing vision statements
- Program sponsor documents containing vision statements
- Descriptions of DoD and service-specific net-centric and SOA initiatives
- Technical information about net-centricity and SOA
- Information about relevant technology efforts

- Net-centric and SOA migration ***lessons learned***
- Enterprise and program-specific architecture information

There may be existing program migration documentation that is relevant to the migration to net-centricity and SOA. The documentation may be incomplete, out of date, or not directly address net-centricity or SOA, but it is still potentially a critical input to the migration planning process.

Use the Joint and Service-specific vision and strategy statements to guide the migration. While often high-level, these statements provide essential enterprise-level context from which the program can develop its own overall vision as well as its vision for migration to net-centricity and SOA.

There are many existing DoD and Service-unique initiatives that aim to enable **Network Centric Warfare** (NCW). These initiatives may impose specific technical and other constraints on a program and may contain relevant guidance. Incorporate activities specific to any of these initiatives into the overall migration planning and execution. While these initiatives provide direction and help to programs, the responsibility for the migration of a program remains with program management. **ASD(NII)** and DISA are valuable sources for information on relevant DoD initiatives. Service **Chief Information Officers** (CIOs) and acquisition agencies are sources for information on service-specific initiatives. This information is useful throughout the migration planning process.

Learn the principles and benefits of Network Centric Warfare (NCW) and SOA-based approaches to NCW. Use NESI and its references as needed to obtain background information. Identify potential benefits of NCW to specific program users and how the principles of SOA would enable the NCW implementation. NESI and its references aid in applying these enabling technologies.

There are many technology initiatives - both within the DoD and within industry - that may be relevant. These may not only directly affect the migration but also may affect the technical standards upon which the migration is based. Collect information on relevant technology initiatives and standards efforts.

Both the DoD and commercial industry are starting to collect and document experiences in migrating to net-centricity and SOA. Collect information on migration experiences.

Document the As-Is Architecture

Collect and document architecture information describing baseline inventories and other items relevant to migration. This information should be at appropriate levels of detail for the intended use and should align where needed with corresponding architecture information from outside of the program's scope (e.g., enterprise-level architecture products and architecture products from other programs). This information includes the following:

- List of key stakeholders and their needs
- Key operational processes (to include key activities and their information exchanges, time-dependent specifics such as states and events)
- Established enterprise integration communities (e.g., **Communities of Interest**, Working Groups) and community standards
- Data assets, **DoD Discovery Metadata Specification** (DDMS) records, XML schemas used in the program, registries used. (Service-specific implementations of the DoD Net-Centric Data Strategy may have specific guidance on compiling the initial data-related aspects of the architecture.)
- Services produced and services consumed
- External interoperability points, public interfaces, dependencies, responsible points of contact
- Applications, hardware systems, Node infrastructure, fielded **software components**
- Roles of the applications, hardware systems, Node infrastructure, and fielded software components in operational processes

- Reusable software components, architectural patterns and other Net-Centric solutions
- Management measures established to govern services
- Enterprise **COTS** licenses

Best Practices

- BP1838: Develop as-is architecture artifacts to support the migration to **net-centricity** and SOA.

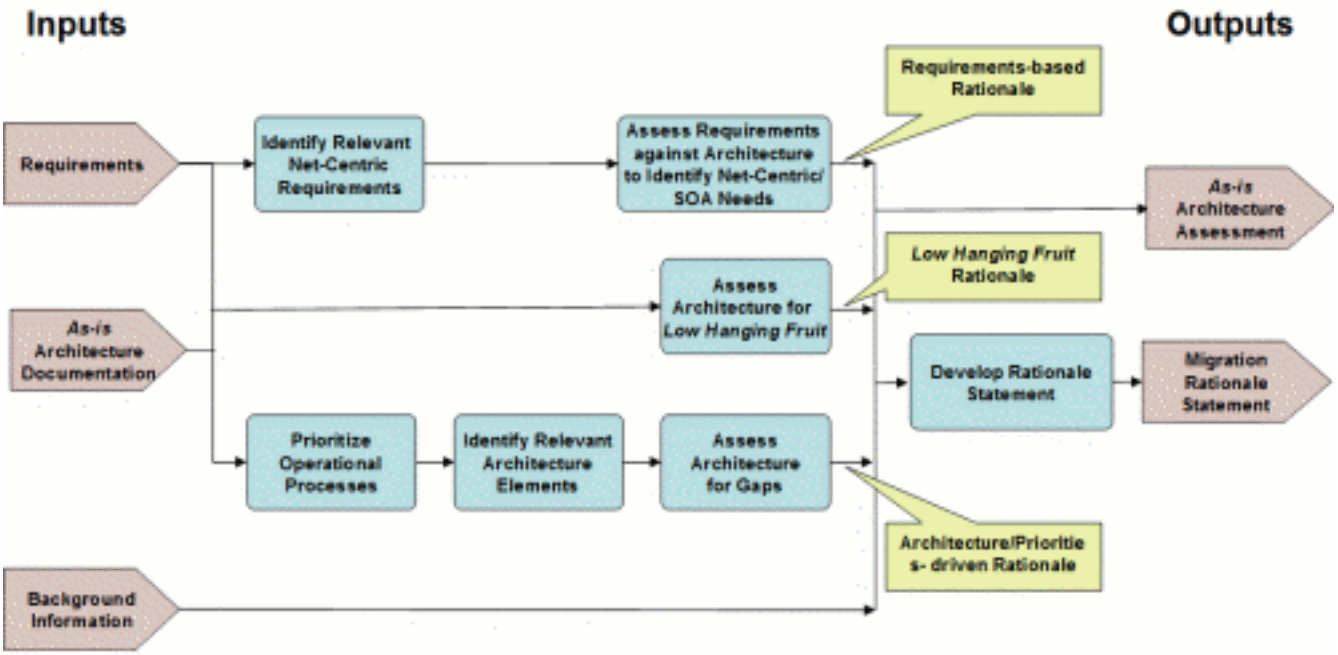
P1206: Assess Migration Needs

A key step in developing the migration plan for a program is to determine the needs-based rationale for the migration. The program identifies the needs as part of performing a trade-off analysis and developing a business case to support the migration. The program then continuously revisits the business case as the program gains experience or as conditions change. A particular program may not need to migrate to any greater degree of net-centricity or services-based. At the other end of the spectrum, it may need to eventually become fully net-centric and services-based. For example, after analyzing various trade-offs associated with migration options, it may be that exposing services is not the only, or even the best, solution. In cases where there is no plan to implement services, implement other net-centric improvements that provide the foundation for better interoperability, flexibility and robustness of the architecture (e.g., n-tiered and componentized architecture, migration to **IPv6**, adherence to enterprise Data Strategy).

Perform the following actions to assess migration needs:

- Assess as-is requirements
- Assess as-is architecture
- Develop migration rationale statement

The following figure depicts these activities in more detail.



The rationale for the migration can stem from a number of sources as shown on the left side of the diagram. The most clear-cut rationale is that there is an explicit requirement that can be directly traced to a net-centric improvement. At the other end of the spectrum are "good ideas" - many of which come from operational experience or enabling technologies. In the middle are general DoD mandates to migrate to net-centricity.

Note that there are numerous sub-steps in the process that call for assessing the architecture - e.g., assessing the architecture against requirements, assessing the architecture for so-called "low hanging fruit," and assessing the architecture for gaps (and overlaps). A common architecture assessment methodology can serve as the basis for these different types of assessments. See the Net-Centric and SOA Assessments perspective for a more detailed discussion about assessment.

In many cases, the program can develop alternative target architecture(s), or elements of it, in parallel with assessing migration needs solutions (see the Develop Alternative Target Architectures perspective).

Best Practices

- BP1839: Perform a business case analysis to support the migration to net-centricity and SOA.

P1209: Assess As-Is Requirements

In this step, the program looks at existing requirements for the program to determine if a case can be made to migrate to net-centricity or SOA. While the focus of this effort is the analysis of the problem, some of the activity contributes directly to developing solutions (see the Develop Alternative Target Architectures perspective).

Multiple types of existing program requirements could create an opportunity to migrate to net-centricity and become the source for migration requirements:

- Explicit program requirements for net-centric improvements
- Program technical requirements that could be fulfilled through making systems more net-centric
- Program requirements in war fighter capability improvements that could be achieved through net-centricity and SOA
- External (primarily joint) mission requirements for interoperability
- Any maintenance improvements and fixes
- Baseline requirements changes

In addition, the program and its systems could be subject to requirements imposed by NCW-related initiatives (see Migration Concern: Conformance with Relevant DoD Initiatives) and constraints imposed by technical standards developed in the enterprise community. Consider these requirements along with other sources for migration requirements in formulating the migration rationale for the program. Identify these requirements early on as part of the ***Compile Migration Background Information*** activity.

The strongest rationale for the migration to net-centricity or SOA stems from stated program requirements. These requirements are generally stated to varying degrees of specificity. Net-centricity and SOA may not be stated as explicit direct requirements but rather as derived technical requirements.

When technical requirements are broadly stated (e.g. "make systems more net-centric," "implement SOA"), assess the requirements against the as-is architecture to formulate more specific migration needs. These needs could be in terms of the following areas:

- operational (e.g., change an organization, change a process)
- technical (e.g., provide a piece of materiel)
- policy (e.g., use the materiel in a particular way, manage the acquisition of services in a particular way)

Carefully distinguish between threshold and objective requirements. Threshold requirements are generally candidates for early increments. Objective requirements may be candidates for early increments when the business-case analysis supports their inclusion.

Applying a metrics-based net-centric and SOA assessment method (see Net-Centric and SOA Assessments) helps to quantify the migration needs (and also monitor progress during the migration). It may not be necessary to assess all systems for net-centricity, but rather only those that are relevant to the identified program requirements.

The output from this activity is the formulation of net-centric and SOA needs based on the analysis of the existing program requirements.

Best Practices

- BP1840: Identify opportunities to apply the principles of net-centricity and SOA throughout the course of the program.

P1210: Assess As-Is Architecture

The analysis of the existing requirements is the integral part of deriving the net-centricity and SOA migration needs, but a program can also derive the migration rationale directly from an analysis of the current state of the architecture. The documentation of the as-is architecture serves as the primary input to this activity. This analysis can use one or more complementary approaches:

- Identify an easily obtainable migration objective, make a solid business case for it, and establish the operational priority for accomplishing the objective.
- Methodically assess the as-is architecture from the perspective of operational priority to identify gaps and overlaps. This includes, but is not limited to the following:
    - Identify and prioritize key operational processes that support mission capabilities
    - Identify what systems (both mission and infrastructure) support those capabilities
    - Identify any special requirements for high performance, security, reliability, availability, or real timeIdentify any information stored in the program's systems that would be beneficial to the enterprise or other elements of the operational process and identify the relevant data assets that may be exposed as services (see the Net-Centric Data Strategy perspective for a more in-depth discussion on data strategy)
    - Identify and prioritize gaps in meeting measures of performance and measures of effectiveness associated with the operational processes
    - Identify unnecessary system redundancy
- Identify deficiencies in the as-is architecture based on its net-centric and SOA assessment. See the Net-Centric and SOA Assessments perspective for a more in-depth discussion about approaches that are available for measuring the maturity of net-centricity and SOA adoption.

The technical guidance throughout NESI can aid in this assessment. In developing the rationale for net-centric or SOA migration, it may not be necessary to assess the entire architecture for net-centricity, but only selected parts based on operational priority. Note that regardless of the source for the migration rationale, metrics-based assessment tools can be very useful. Use a consistent methodology to develop the initial rationale, make the baseline assessments, and assess planned and actual progress throughout the migration.

Start to plan the migration as a side-effect of doing this assessment. Make the vision for the migration to net-centricity and SOA explicit. This migration vision should be consistent with both the program's and the enterprise's overall vision. Identify preliminary potential net-centric improvements to systems that support the desired net-centric and SOA-based capabilities. Analyze existing data access mechanisms, both service-based and non service-based, for potential improvements. Make preliminary estimates of the benefits to be derived from those improvements. Consider the constraints imposed by any special requirements as well as cost, schedule, and risk. This assessment is a subject to trade-offs and business case analysis in later stages of the migration planning process.

Best Practices

- BP1841: Involve key stakeholders in the assessment of the as-is architecture in preparation for the migration to net-centricity and SOA.
- BP1847: Use the same assessment methodology to assess the as-is architecture, define the target (to-be) architecture for each migration increment, and assess migration progress at the end of each migration increment.

P1211: Develop Migration Rationale Statement

A Migration Rationale Statement documents the motivation for the migration. Consolidate the assessment of the requirements and the architecture into a Migration Rationale Statement which includes the following:

- Migration Vision
- Assessment of the as-is requirements
- Assessment of the as-is architecture
- Identification of migration needs (in addition to any documented migration requirements)
- Identification of issues
- Identification of potential system improvements (to include pedigree)
- Statement of expected benefits
    - Direct benefits - capability improvements for the warfighter and support elements
    - Financial and management benefits (e.g., through reuse, elimination of functionality overlaps, agility in regards to future change)
- Preliminary considerations about cost and risk

Best Practices

- BP1842: Formally document the migration rationale to support the migration to net-centricity and SOA.
- BP1843: Obtain consensus among all key stakeholders on the rationale for the migration to net-centricity and SOA.

P1207: Plan Migration

Most migration efforts require a formal migration plan. A migration plan helps to synchronize technology improvements, overall mission/business capability improvements, and the changes in the business practices enabled by SOA while making explicit the return on investment (ROI). A migration plan, while more strategic in nature, serves as the basis for detailed project plans for specific near term migration increments.

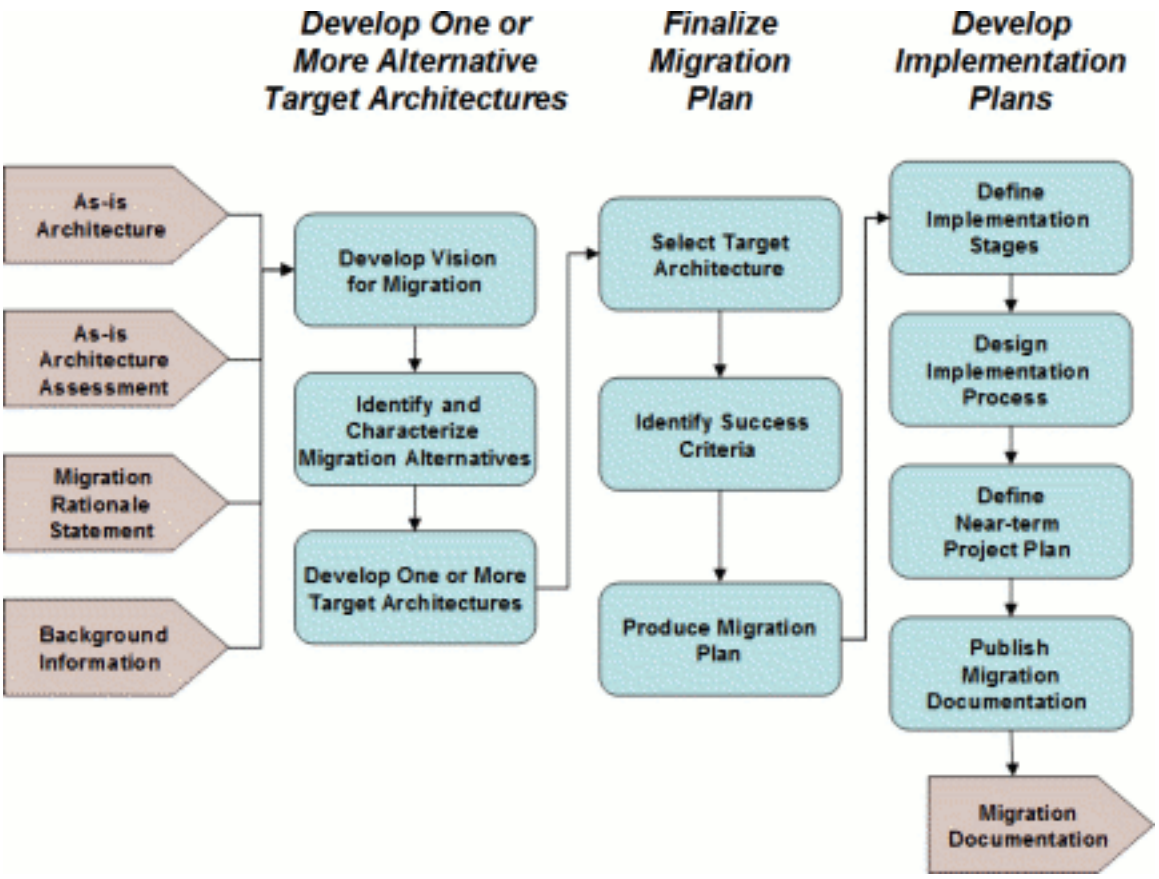
A migration plan is based on the ***Migration Rationale Statement*** (see the Develop Migration Rationale Statement perspective) which indirectly represents requirements, the as-is architecture documentation, and the background information gathered early in the planning process.

To plan the migration, perform the following actions:

- Develop Alternative Target Architectures
- Finalize Migration Plan
- Develop Implementation Plans

These activities help to answer the questions of ***what*** needs to be done and ***how*** this will be implemented (the target architecture). The output is a set of migration plan documentation that includes an updated vision and goals statement, a high-level migration plan, and implementation plans for the increments of the migration. In many cases, development of the target architecture(s), or elements thereof, can occur in parallel with the actions included in the Assess Migration Needs perspective.

The following figure depicts these activities in more detail. Note that many of these activities may overlap in time.



P1212: Develop Alternative Target Architectures

The program explores multiple options, or alternative target architectures, as part of the business case analysis in support of a net-centric migration. The program-specific vision for the migration should guide the target architecture. This, in turn, should be driven by higher-headquarters vision and strategy documents, service-specific vision documents, and the program's overall vision.

A program may choose to develop more than one alternative target architecture to varying degrees before selecting a single target architecture to pursue in the migration (see the Finalize Migration Plan perspective). The alternatives might be slight variations from each other or represent significantly different solution approaches depending on the program needs and the resources available to develop the architectures. These alternative architectures should conform to a common reference architecture that identifies predetermined functions and their interfaces.

Alternative target architectures are generally a step toward migration; however, they may not fully realize net-centric or SOA principles but will rather reflect the program's specific requirements and environment. Represent the alternative target architectures as time-stamped and version-stamped architecture data. This permits viewing the future architectures as they progress over time and as alternative "branches" of development are explored. Time-phase the alternative target architectures to support the development of migration increments.

The program takes initial steps towards developing the vision for the migration and one or more target architectures during the Assess Migration Needs activity. In the Develop Alternative Target Architectures activity, the program refines those alternative target architectures. The program identifies and characterizes the alternatives by examining migration needs and options and further develops the architectures by analyzing multiple factors associated with those options. Note that while a program might develop multiple alternative target architectures, these architectures might vary greatly in detail and degree of formal documentation.

Perform the following activities to develop these alternative target architectures:

- Identify potential net-centric improvements to systems that support key mission capabilities.
- Analyze existing data access mechanisms, both service-based and non service-based, for potential improvements.
- Consider the constraints imposed by any special requirements such as performance, security, reliability, availability, or real time.
- Consider the impact of DoD or Service-unique net-centric initiatives.
- Where appropriate, identify net-centric migration alternative approaches to SOA-based implementations due to special requirements and other considerations (see the Migration Patterns perpsective for additional insight)
- Focus initial migration efforts on applications that have known issues. Migrate applications that are performing well later in the migration effort. This may help reduce risk from the changes in the systems due to migration.
- Focus initial migration efforts on the implementation of data services to provide a fast return on investment.
- Plan to migrate to net-centricity even in cases when there is no plan to implement services. A focus on net-centric implementations provides a foundation for future change and may support the future implementation of services. Such efforts may include migrating to n-tiered and componentized architecture, migration to **IPv6**, adherence to the **DoD Net-Centric Data Strategy**, etc.
- Determine the target pedigree of any potential service. A clear understanding of the target pedigree of a potential service (e.g., local consumers, **Community of Interest** (COI) or enterprise-wise consumption, etc.) can help to identify potential issues and focus the use of resources on high-priority activities.

- Identify and document **Systems Development Lifecycle** governance, and management issues associated with a service. A clear understanding of issues that will not be addressed is important for managing expectations and allocating resources. Issues that will not be addressed are potential inputs to future migration activities.

- Identify and document requirements and issues that will not be addressed during migration. A clear understanding of issues that will not be addressed is important for managing expectations and allocating resources. Issues that will not be addressed are potential inputs to future migration activities.

The program may link operational deficiencies to architectural net-centricity deficiencies and the implementation of new net-centric features may enable new operational capabilities. Work with the operational community to identify explicitly such migration opportunities.

To help in the selection of a single target architecture, characterize one or more alternative target architectures along relevant dimensions that may include the following:

- Operational context
- Relevance to program requirements
- Short/long-term benefit
- Extent of solution (e.g., full or partial requiring further action)
- Degree of net-centric improvement (e.g., improved net-centric assessment scores)
- Degree of SOA adoption advancement
- Creation of new dependencies, addition of stakeholders
- Relevance to DoD net-centric initiatives
- Subject to enterprise community standards/constraints
- Performance, security implications
- Testing requirements, ability to test
- Cost (including cost of software fielding and maintenance, required personnel training, etc.)
- Risk

Assess the requirements and needs against the as-is and alternative target architecture(s) to make explicit linkages between them and potential net-centric and SOA-related architectural improvements.

Technology opportunities may also influence the alternative target architecture(s). This includes inputs from DoD and commercial sources and may influence the structural elements of the architecture as well as the standards and principles that guide the design of the system. Assess these opportunities and incorporate them where appropriate.

Best Practices

- BP1844: Develop a vision statement for the migration to net-centricity and SOA.
- BP1848: Develop one or more target architectures for the migration.

P1213: Finalize Migration Plan

The program selects a target architecture based on a business case analysis and assembles the migration plan documentation. The program defines migration goals, obtains stakeholder consensus, and then publishes the long-term high-level migration plan that will drive the development of detailed short-term project plans for the migration increments.

Use the detailed characterization of the alternative target architectures (see the Develop Alternative Target Architectures perspective) to select one that best meets the enterprise's vision and the program's needs, requirements, cost, schedule, and risk constraints.

Formulate specific goals for the migration. Even though the goals are expected to evolve as various influencing factors change, this will help keep the target migration benefits in focus during the entire process. This may also provide a good basis for ***lessons learned*** as plan revisions occur. Use the same goal setting methodology when revising migration goals during migration execution. See the Migration Patterns perspective for a discussion about various technical migration goals based on the initial state of a system.

The migration plan includes the following elements:

- ***Migration Rationale Statement*** (to include the ***Migration Vision Statement***)
- Description of the time-phased target architecture
- Timeframe for implementation
- Results of the business case analysis, including cost/benefit and risk analysis and immediate and potential, longer-term, benefits expected from the migration
- Analysis of trade-offs that led to accepted solutions
- Description of technical solutions adequate for reuse
- Stakeholders, responsibilities, dependencies
- Test strategy (to include criteria to use to measure the success of the migration)
- Considerations for an implementation process (e.g., agile methodology, existing processes used by contractors)

Best Practices

- BP1835: Develop a formal migration plan to support the migration to **net-centricity** and SOA.
- BP1836: Obtain consensus on the migration plan from all key **stakeholders**.
- BP1837: Update the **net-centric** and SOA migration plan in an iterative manner as the program gains migration experience and conditions change.

P1214: Develop Implementation Plans

The final stage of migration planning is to build detailed project plans for the migration increments. Define the initial (or current) increment in detail and define subsequent increments to the extent feasible. To plan the implementation of the target architecture, perform the following actions (some of which can occur in parallel):

- Define implementation increments
- Design implementation process
- Create near term project plan
- Compile and publish plan documentation

Use the ***Migration Vision Statement*** to identify one or more implementation increments leading to the full implementation of the target architecture. Use the following parameters to identify each increment:

- Implementation increment timeframe
- Deliverable goals and associated benefits, including architecture evolution analysis (this may include assessment of net-centricity and analysis of SOA state in the beginning and the end of the increment using assessment tools; see the Net-Centric and SOA Assessments perspective).
- Specific identification of the needs that will be satisfied and to what degree they will be satisfied
- Cost projections, cost/benefit analysis for the increment
- Analysis of trade-offs that led to accepted solutions, including cost/benefit and risk analysis
- Test strategy
- Description of technical solutions adequate for reuse
- Responsibilities, dependencies
- Exit criteria

Design an implementation process that fits with the established acquisition framework for the program. In many cases, it will be sufficient to identify elements of the existing processes that are relevant to the net-centric migration and enhance them where necessary to meet the objectives of the migration. Ensure the process includes mechanisms for collaboration with users and for updates to the plans.

Create a near-term project plan for the next funded increment(s):

- Compile a list of key concerns (see the Critical Migration Concerns perspective) and include tasks associated with each of them in the migration increment project plan
- Update the list of key concerns and tasks as migration executed (see the Execute, Monitor, and Control Migration perspective)
- Create work packages for the near-term migration project.

The product of the migration planning activities is a set of migration planning documentation that must be maintained and updated throughout the migration (see the Execute, Monitor, and Control Migration perspective). Publish this documentation for the benefit of other programs and enterprise-level personnel (with appropriate security considerations). Include the following information:

- The ***Migration Vision Statement***
- Results of any baseline net-centric and SOA assessment
- Explanation of how this net-centric migration incorporates or coordinates with DoD **NCW** initiatives
- Definition of migration increments
- Description of the implementation process
- High level description of the near term project
- ***Lessons learned*** to-date (both good and bad)

Obtain **stakeholder** consensus with the detailed project plans and publish the complete set of migration documents.

Best Practices

- BP1846: Involve key stakeholders in the development of the implementation plan increments.
- BP1845: Consider key enterprise-level concerns when planning and executing a migration to net-centricity and SOA.

P1208: Execute, Monitor, and Control Migration

The overall migration process is generally iterative and interactive. Migration-related activities overlap in time and often do not occur in a strict order. All of the steps are interrelated and have numerous feedback paths. A key feedback path is from the execution of migration increments into the evolution of the overall migration plan as well as the development of the detailed project plan for the next migration increment in a spiral fashion.

After the project plan for a migration increment is completed, approved, and the migration is started, use project monitoring and scope/plan adjustment methods that have been established for the program (e.g., Earned Value Management, Agile Methodology). Update the project plan at least once per migration increment.

Maintain the list of Critical Migration Concerns and related actions. Review the list for completeness and effectiveness. Make adjustments in future planning activities to include modifying both the list of concerns and the program actions taken as a result of the concerns.

The end-of-increment state becomes the current state of the program going forward. To answer the question "where are you now?" and to be able to analyze the progress to this point, it is necessary to assess the results of migration performed in the completed migration increment. This includes assessing progress in satisfying explicit net-centric requirements and performing a net-centric and SOA assessment (see the Net-Centric and SOA Assessments perspective).

Compare the results of migration performed in the completed migration increment with the projected outcome resulting from the migration planning process. This helps to determine how much work remains and to identify the factors that contributed to any deviation and take them into account in future migration planning. Based on the analysis of the achieved versus projected outcomes and the factors that contributed to any deviation, formulate and record any "lessons learned" (both good and bad).

Programs need to maintain the as-is architecture documentation continuously. At a minimum, update the as-is architecture documentation at the end of each of the migration increments. Changes could result from migration already performed as well as from external factors (e.g., changes in the external dependencies, evolution of enterprise technical standards).

Requirements for the program may change during the increment execution and old requirements may also present new opportunities for net-centric migration (e.g., because of new developments in technology, standards and enterprise environment and newly discovered problems in fielded systems). Re-analyze and re-prioritize requirements in order to proceed with further migration planning.

DoD net-centric initiatives may affect the program's net-centric migration. Stay informed of progress related to these initiatives and update related migration tasks to reflect them (either by direct inclusion or coordination; see the Migration Concern: Conformance with Relevant DoD Initiatives perspective).

The program may need to revise migration goals and corresponding work packages when the migration environment changes or when motivated by ***lessons learned*** (either directly or from an external source). Reexamine the environment periodically and revise the goals accordingly. Also, review and update the target architecture and migration vision documentation periodically based on a review of migration progress and lessons learned.

P1201: Migration Patterns

Each migration is unique; however, there are some general migration ***patterns***, or templates, that may be useful when developing and executing a migration plan. These patterns describe approximate start and end points of typical migrations and the migration steps associated with them. A program may be able to leverage one or more of these patterns to the extent that its situation matches that of the identified patterns. In such a case, program personnel will need to adapt the relevant patterns to the program's specific circumstances.

The starting points present various current levels of net-centricity and infrastructure investment. The description of the steps include considerations of some of the trade-offs associated with them. The end points identify future migration options. The patterns help program personnel perform the business case analysis for the program's migration. While multiple patterns may apply to a specific program, the patterns provide an aid for determining potential end state architectures (both near- and far-term) as well as specific transformation steps for achieving those architectures.

Leverage the migration patterns when doing the following:

- Developing migration business cases.
- Developing alternative target architectures.
- Developing migration plans.
- Communicating with peers.

Migration Starting Points

The NESI Part 3 migration patterns are grouped by one of three starting points:

- Non-Componentized
- Layered and Componentized
- SOA-Enabled

P1215: Non-Componentized Migration Starting Point

This starting point is characterized by the use of proprietary **APIs** and little or no Web access.

Initial Net-Centric State

- No connectivity
- No interoperability at the information level (no standard interface is available)
- Not layered or componentized

Initial SOA Maturity

- No implementation of services or supporting infrastructure

Patterns Associated with this Starting Point

- Exposing Functionality through Non-Standard Interfaces
- Wrapping Legacy Code into a Service
- Re-Implementation

P1218: Pattern: Exposing Functionality through Non-standard Interfaces

As an initial step to greater degrees of net-centricity and SOA - and to satisfy mission requirements quickly - it is possible to expose functionality or **data** from a **legacy system** by exposing an **interface** to its implementation. The interface does not necessarily need to be in a standard format in the early stages of the migration (if ever). One way to do this is to use simple client-server technology and existing infrastructure. In some cases, this approach might require the use of an enterprise-level translation service to integrate with other systems.

While this is a straightforward approach, if done properly it can aid future net-centric modernization. This is especially important if the functionality is part of a key operational process.

This approach can also be used in parallel with other technology solutions. For example, **FTP** can be used in parallel with **messaging** to transport large chunks of data (while messaging alone may not be well-suited for that).

Benefits

- Critical data or function is available to other systems quickly
- Reduced cost
- Low risk
- Provides initial integration experience with other systems
- May prepare the functionality for easy conversion into a service-oriented implementation

Net-Centric Outcome

- Critical connectivity may be attained
- System is generally not interoperable at the information level due to use of non-standard interfaces
- Connectivity could have some net-centric elements (e.g., **IPv6**, asynchronous messaging)
- System's internal architecture might remain mostly (or entirely) not net-centric
- System's internal architecture might remain mostly (or entirely) not layered
- System's internal architecture might remain mostly not componentized (except possibly for specific functionality exposed through interfaces)

SOA Maturity of the Outcome

- No implementation of services or supporting infrastructure, but the outcome can serve as the basis for a future migration to SOA
- Provides initial collaboration experience with other systems

Further Options

- If the exposed entity will remain an important element of mission or business capability, there are several options for its further migration toward net-centricity:

- Turn it into a **software component** by separating it from its internal implementation, if it has not already been done
- Convert the interface to XML and use a standard XML schema
- Wrap the exposed interface into a service (see the Pattern: Wrapping Legacy Code into a Service perspective)

- Consider porting the application to a multi-tiered and componentized architecture. This would typically be done along with **Web-enablement**, which provides a natural environment for developing more secure, robust and scalable Web Services. See the Layered and Componentized Starting Point perspective for more information on migration options for Web-enabled systems.

Best Practices

- BP1849: Delay the decoupling of interface from implementation until the migration to a standard interface.

P1219: Pattern: Wrapping Legacy Code into a Service

Wrapping or encapsulating legacy code allows the use of the legacy code as a service without porting or re-implementing it (which could be more costly and involve substantial risk). SOA can bridge architectural gaps between **legacy systems** and between legacy and new net-centric systems.

Wrapping legacy code often requires some internal code modification, however. The level of the invasiveness depends on factors such as the following:

- the granularity and structure of the legacy code (e.g., the degree to which it lends itself to exposure as a service)
- the degree of the legacy code's dependence on its native environment (sometimes the code needs to be extracted and placed into a separate compilation unit)

Wrapping does not, however, solve the problem of dealing with poorly structured legacy code which might remain inside the service and which might be costly to maintain. A further option is to refactor the service implementation code (see the Pattern: Re-implementation perspective).

At first, the SOA-based messages could use simple client-server protocols for information delivery (e.g., **FTP**) that do not require an investment in expensive infrastructure. At a later migration phase, the same message could be converted into a **Web service** and communicated over **HTTP** (see the Exposing Web Services pattern) or an **Enterprise Service Bus** (ESB).

Benefits

- Allows reuse of core legacy software assets in SOA standard environment quickly
- Provides a viable option for moving critical legacy assets to SOA due to higher risk associated with porting/re-implementing the code

Net-Centric Outcome

- System is connected
- System is interoperable at the connectivity point in question
- Net-Centricity of the connectivity model varies based on the delivery method of the service
- Wrapping into a standard service provides decoupling between the service interface and its internal implementation
- Internal implementation of the service (underneath the service shell) remains not net-centric
    - System's internal architecture may not be layered (a drawback of this is that if the newly created service uses a tightly coupled database, this service may not scale well when the number of users increases)
- A service is a component; the rest of the system's architecture remains not componentized
- Changes systems to SOA-enabled

SOA Maturity of the Outcome

- The resulting service can be part of SOA at any phase of maturity, but wrapping is typically used as a method to expose initial services in the Early Learning phase of SOA adoption

Further Options

- Expose more services
- Gain experience by making the service available to more users and establishing **Service Level Agreements**
- Improve the existing service. Elevate its scope (e.g., number of users, adding remote users, establishing quality of service contracts). Make **Systems Development Life Cycle** process more rigorous
- Make the service a part of an **orchestrated** flow
- Implement automated **discovery**

Best Practices

- BP1850: Use service design guidelines and best practices to convert an interface into a service.
- BP1851: Focus wrapping efforts on key operational processes.

P1220: Pattern: Re-Implementation

Consider legacy systems that implement an important mission-specific functionality for re-implementation. This is especially true for long-lived functionality. Layer the new implementation which may be **component**-based or fully service-oriented. (A layered and component-based architecture would create a foundation for future SOA implementation through the flexibility of the "separation of concerns" inherent in such architectures.) Such an effort is more costly and generally requires a longer-term investment, but it also tends to provide considerable long-term return. The cost includes code **refactoring** and the investment in infrastructure to support the layered architecture. The motivation for re-implementation is stronger when elements of this infrastructure (e.g., Web and application servers, databases, networking, messaging middleware) are already available.

A requirement to **Web-enable** a system presents a classical opportunity to move to the layered architecture because refactoring the code into components typically happens during porting of the application to a modern Web infrastructure (at a minimum: **Web server**, or an **application server** with a Web server). Modern Web infrastructure provides additional benefits like improved performance, scalability, availability, security, etc., through the advent of its internal architecture. It allows internal or external clients to connect, and it enables implementation of **Web services** in the future.

Decide at which point to migrate to a multi-tiered architecture. For example, if there is no evident need to expose or consume services outside of the program in a near time, while a robust and scalable Web-enabled integration internally would provide considerable benefits to the user, it would make sense to start with Web-enabling of the system and to re-engineer the architecture as multi-tiered and componentized at the same time.

Benefits

- Allows for more rapid and agile development, improvement, troubleshooting, and reuse of the code
- Prepares for public exposure of interfaces, including services

Net-Centric Outcome

- If Web-enabled (typical re-implementation outcome), Web clients may be connecting to the system via **HTTP** or **HTTPS**; Really Simple Syndication (RSS) feeds might be available
- Componentized and layered application architecture is a basic tenet of net-centricity
- If Web-enabled, this changes systems to the Layered and Componentized migration starting point

SOA Maturity of the Outcome

- Minimal change from initial state unless services are implemented

Further Options

- Expose components that are aligned with operational processes as services.
- Web-enable applications (see the Layered and Componentized migration starting point).

Best Practices

- BP1852: Align software components with operational processes.
- BP1853: Apply applicable service development best practices to software component development.

P1216: Layered and Componentized Migration Starting Point

This starting point is characterized by the use of a layered, component-based, **Web-enabled** architecture.

Initial Net-Centric State

- Componentized and layered application architecture
- Web clients connect to the system primarily via **HTTP** or **HTTPS**

Initial SOA Maturity

- No implementation of services
- Infrastructure enables development of **Web services**

Patterns Associated with this Starting Point

Consider the following patterns for this starting point:

- Exposing Web Services
- SOAP over JMS

P1221: Pattern: Exposing Web Services

Exposing **Web services** is a natural next step from **Web-enabled** applications. Typically, "it is another layer on top of the infrastructure that already exists" [R1208] and the infrastructure is often multi-tiered. For such systems, an additional return on investment (ROI) is realized because **COTS** infrastructure products (e.g., **application servers**, **Web servers**) normally provide Web services extensions as part of regular upgrades at no additional cost.

Often a combination of technologies proves to be most effective. For example, existing Really Simple Syndication (RSS) feeds could provide a publish/subscribe notification of new data that can then be fetched by invoking a Web service at a specified URL. Initially, some data could be delivered via RSS before full implementation of a Web service is available.

Benefits

- System becomes SOA-enabled (see the SOA-Enabled Migration Starting Point perspective)

Net-Centric Outcome

- System further enriched in the areas of **services** and **data**

SOA Maturity of the Outcome

- Various, depending on degree of data exposure (see the Phases of SOA Adoption perspective)

Further Options

- Expose more services
- Gain experience by making the service available to more users and establishing **Service Level Agreements**
- Improve the existing service, elevate its scope (e.g., number of users, adding remote users, establishing quality of service contracts), and make **Systems Development Life Cycle** process more rigorous
- Make the service a part of an **orchestrated** flow
- Implement automated **discovery**
- For asynchrony, publish/subscribe; for greater reliability and scalability use **MOM** or **ESB** approaches as an alternative to **HTTP** transport, including use of **SOAP** over **JMS** (see the SOAP over JMS pattern)

Best Practices

- BP1851: Focus wrapping efforts on key operational processes.
- BP1850: Use service design guidelines and best practices to convert an interface into a service.

P1222: Pattern: SOAP over JMS

JMS provides an underlying messaging middleware as an alternative to **HTTP** transport for **SOAP** messages. JMS is natively a Java interface standard and is intended for Java applications (although some **COTS** and open source solutions are available to bridge it with the **.NET** framework). The underlying implementations of the interface are vendor-specific and therefore require bridging so that two JMS communicating peers from different vendors can interoperate. Since interoperability standards are lacking for SOAP over JMS, JMS is often used locally and bridged to **HTTP** at the boundary of the node (combination of the Exposing Web Services and SOAP over JMS patterns).

Benefits

- System becomes SOA-enabled (transitions to the SOA-Enabled migration starting point)
- Greater reliability and scalability than in the Exposing Web Services pattern
- Asynchronous and publish/subscribe modes of messages delivery are available

Net-Centric Outcome

- System further enriched in the areas of **services** and **data**
- Additional connectivity over messaging middleware (could be an **enterprise service bus**)

SOA Maturity of the Outcome

- Various, depending on degree of data exposure (see the Phases of SOA Adoption perspective)

Further Options

- Expose more services
- Gain experience by making the service available to more users and establishing **Service Level Agreements**
- Improve the existing service, elevate its scope (e.g., number of users, adding remote users, establishing quality of service contracts) and make the **Systems Development Life Cycle** process more rigorous
- Make the service a part of an **orchestrated** flow
- Implement automated **discovery**

Best Practices

- BP1854: Use **SOAP** over **JMS** only when implementation interoperability is not the main driving factor.

P1217: SOA-Enabled Migration Starting Point

This starting point is characterized by the use of SOA constructs such as a **Web service** infrastructure. Some functionality may be exposed via **services**.

Initial Net-Centric State

- Service-based connectivity with a variety of underlying transports
- System is interoperable to the extent of the use of industry Web services and **COI** standards
- System may or may not be internally layered and componentized

Initial SOA Maturity

- SOA-enabled, at any state of SOA maturity

Migration options for these types of systems depend greatly on how and why the services were created and the SOA maturity in the rest of the enterprise.

Patterns Associated with this Starting Point

Consider the following migration pattern:

- Changing Internal Implementation of the Service

P1223: Pattern: Changing Internal Implementation of the Service

A next step after creating a **service** by wrapping legacy code (see the Wrapping Legacy Code into a Service pattern) could be to improve the internal implementation of the service without changing the external interface. The improvements may include changing the internal architecture of the application to make it more layered and componentized (e.g., porting to a standards-based **application server**).

Benefits

- Improved underlying service implementation resulting in improved performance, security, and service availability
- Architecture is more flexible for exposing new services in the future or changing the existing services
- Componentized code is generally less expensive to maintain than legacy code, and it may enable reuse
- Componentized code may enable reuse

Net-Centric Outcome

- Improved layering of the architecture

SOA Maturity of the Outcome

- No change in the SOA maturity

Further Options

- Expose more services
- Gain experience by making the service available to more users and establishing **Service Level Agreements**
- Improve the existing service, elevate its scope (e.g., number of users, adding remote users, establishing quality of service contracts) and make the **Systems Development Life Cycle** process more rigorous
- Make the service a part of an **orchestrated** flow
- Implement automated **discovery**

P1202: Critical Migration Concerns

A successful migration to net-centricity entails addressing multiple integration and management concerns early and continuously throughout the migration. These are, for the most part, traditional software development concerns that acquire a new **enterprise** perspective due to the focus on net-centricity and SOA. It is no longer a "design in isolation, integrate later" approach.

Many different program-related management and engineering personnel must come together with this fresh perspective to ensure success:

- Project and product managers
- Acquisition and contracting specialists
- Enterprise and systems architects, including operational architects
- Application and enterprise integrators, including infrastructure implementers
- Service/application developers
- Technical standards developers
- Governance policy developers
- Governance policy enforcers

These stakeholders can use the following list of key concerns as the nucleus of a checklist when planning and executing the migration. The list emphasizes the importance of several areas. Note that some concern areas overlap in scope and may also overlap with other guidance in NESI. Each concern area provides a set of general program actions as guidance for developing program-specific mitigation actions. These actions depend on the specific program environment and are not covered here in detail. In addition, programs will need to update their migration plans as they address these concerns and as the program and enterprise environments change.

Migration Concerns

The key concerns that a program's stakeholders must consider during migration include the following:

- Focus on Warfighter Needs
- Conformance with Relevant DoD Initiatives
- Conformance with Net-Centric Technical Tenets
- Management Issues for Exposed Functionality
- Infrastructure Dependencies
- System Performance
- Security
- Cost and Benefit Tracking
- Risk Management
- Test in an Integrated Environment

- Migration Plan Maintenance
- Architecture Documentation Maintenance
- Enterprise-Level Migration Knowledge Management

Best Practices

- BP1845: Consider key enterprise-level concerns when planning and executing a migration to net-centricity and SOA.

P1224: Migration Concern: Focus on Warfighter Needs

The overall purpose of a migration to net-centricity is to improve capability. Capability is a complex combination of people, processes, and materiel (in effect, the full range of **DOTMLPF**) that come together to achieve an effect. The as-is and projected future states of capability are represented in architecture products. It is critical to maintain focus on both functional and non-functional warfighter needs throughout the migration. More specifically, maintain focus on improving operational processes throughout migration.

Program Actions

- Define a role within the program to manage the collaboration between the program and the users for the duration of the net-centric migration.
- Obtain user requirements and translate them into technology objectives as appropriate.
- Identify technology objectives that align with net-centricity and SOA (e.g., adaptability, reuse, general interoperability).
- Use agile acquisition and development methodologies to accommodate evolving requirements.

P1225: Migration Concern: Conformance with Relevant DoD Initiatives

There are many existing DoD initiatives that aim to enable **Network Centric Warfare** (NCW) through the net-centric integration of the DoD enterprise. These initiatives provide direction and help to programs in their migration endeavors. These initiatives range from efforts to define enterprise-wide technical strategy to efforts to provide common infrastructure implementations. They include, but are not limited to, such efforts as the DoD Net-Centric Data Strategy (NCDS), Modular Open Systems Approach (MOSA) [R1178] , Net Enabled Command Capability (NECC), Net-Centric Enterprise Services (NCES), the ASD(NII) Net-Centric Checklist [R1177] , and various DoD and **CJCS** directives, instructions and manuals.

These initiatives may impose specific technical and other constraints on a program; Program Managers must be familiar with these various DoD initiatives and take part in those which may be relevant during the migration. Activities specific to any of these initiatives must either be incorporated into the overall migration planning or aligned with other migration activities.

Program Actions

- Maintain awareness of NCW-related initiatives.
- Incorporate the results from NCW-related initiatives into the program migration plans.
- Consider the risks associated with conformance with relevant DoD initiatives (see the Migration Concern: Risk Management perspective).
- Prepare a business case for when deviations from compliance with mandated NCW-related initiatives are necessary (see the Migration Concern: Infrastructure Dependencies perspective).
- See the Net-Centric Data Strategy perspective for NCDS-specific guidance.

P1226: Migration Concern: Conformance with Net-Centric Technical Tenets

The migration to SOA, without an underlying migration to net-centricity, will not automatically achieve the main SOA promise of overall business agility. The migration to net-centricity and SOA requires that systems share a common technical framework to minimize the effort associated with connecting to each other and sharing and protecting information. There are numerous such frameworks that address various levels of technical guidance or that focus on specific technical subject areas (e.g., security, **services**, **data**).

NESI is a body of architectural and engineering knowledge that guides the design, implementation, maintenance, evolution, and use of the **Information Technology** (IT) portion of net-centric solutions for military applications. The technical tenets that are codified in NESI can help programs migrate to net-centricity. NESI does not, however, replace intimate knowledge of program requirements and constraints.

Program Actions

- Develop a technical migration strategy that is consistent with the net-centric strategy and apply appropriate NESI guidance to assist with the migration process.

Best Practices

- BP1847: Use the same assessment methodology to assess the as-is architecture, define the target (to-be) architecture for each migration increment, and assess migration progress at the end of each migration increment.

P1227: Migration Concern: Management Issues for Exposed Functionality

A critical goal of migrating to net-centricity is to expose functionality via **services**. Manage these services carefully during both development and use to ensure that all relevant stakeholders understand the scope and performance of the functionality.

Program Actions

- Coordinate with all relevant stakeholders to develop and maintain detailed descriptions of the services prior to implementation.
- Use detailed service descriptions to identify existing services that are candidates for reuse.
- Coordinate with all relevant stakeholders to document service performance in **Service Level Agreements**.
- Coordinate with all relevant stakeholders to develop requirements for managing changes to Service Level Agreements for services.
- Implement service monitoring, diagnostic, and disaster recovery mechanisms.
- Test service management mechanisms in an integrated test environment that reproduces or simulates the target operational environment (see the Migration Concern: Test in an Integrated Environment perspective).
- Refer to NESI guidance and best practices to support service development.
- Establish an analysis capability that provides meaningful statistics on actual (collected) versus "contracted" (stated in the Service Level Agreement) performance expectations to help guide changes to exposed services.

P1228: Migration Concern: Infrastructure Dependencies

Migration to net-centricity results in dependencies on a net-centric infrastructure that is often outside of the direct control of the individual program. Carefully manage these dependencies to identify cost, risk, and resource issues early on and throughout the migration (see the following migration concerns: Cost and Benefit Tracking, Risk Management, Conformance with Relevant DoD Initiatives).

Program Actions

- Identify and manage dependencies on net-centric infrastructure to include consideration of the following:
  - Architecture (structural, technical standards - both open and closed/proprietary)
  - Resources to perform configuration, tuning, and technical support
  - Resources to manage relationships with **COTS** vendors
  - Cost of porting applications to (and certifying on) new platform
  - DoD infrastructure-specific initiatives

P1229: Migration Concern: System Performance

Migration to net-centricity and SOA is generally motivated by the need for increased information sharing and adaptability to change. This may adversely impact the speed of execution performance of individual systems. However, a related motivation for the migration (from an **enterprise** perspective) is for a collection of programs to contribute to an improvement in some set of capabilities. The stated improvement in capability may include a variety of measures of performance to include such things as overall speed of execution, scalability, quality of output, adaptability to change, operational safety, and reliability of execution. Consider the performance of the individual system within the scope of this larger context.

Program Actions:

- Document system scalability requirements and concrete required performance characteristics relevant to anticipated workloads within the larger enterprise context as expressed in the enterprise architecture.
- Analyze any potential impacts on system performance caused by the migration to net-centricity or SOA early in the design stage.
- Record all considerations associated with performance trade-off decisions.
- Identify key points in the architecture where system performance is critical to the overall success of the system. Use these key points to help monitor migration progress.
- Prototype solutions and conduct performance tests (see the Migration Concern: Test in an Integrated Environment perspective) based on anticipated workloads.
- Use reserve resources for addressing system performance issues. Avoid the use of reserve resources for adding new functionality.
- Use acquisition strategies that allow providers to rapidly respond to changes in workloads and performance needs.
- Review contract award fee strategies to address changing system performance needs and to incentivize meeting or exceeding those system performance needs.

P1230: Migration Concern: Security

Migration to net-centricity often adds new security considerations. Making systems net-centric and interoperable, while very desirable in many respects, can introduce vulnerabilities that require careful consideration. Address security concerns at multiple levels within the architecture while applying an enterprise perspective. The overhead of security mechanisms can adversely affect the performance of functionality exposed on the network. As such, security features are difficult to add on in an effective manner "after-the-fact."

Program Actions:

- Document system security characteristics with respect to the larger enterprise context as expressed in the enterprise architecture.
- Build security measures into designs during the early stages of the migration.
- Analyze security requirements with respect to performance. Identify dependencies and capture appropriate **Service Level Agreement** language to be communicated to security providers.
- Record all considerations associated with security trade-off decisions.

P1231: Migration Concern: Cost and Benefit Tracking

Migration to net-centricity requires continuous and careful attention to cost and benefit trade-offs. These trade-offs are made more difficult, on both sides of the equation, due to the broader scope of the migration.

Program Actions:

- Assess the benefit of the migration based on the improvement in mission capability.
- Identify both short- and long-term benefits at each stage of the migration.
- Apply a consistent cost assessment methodology throughout the migration.
- Assess the cost to acquire infrastructure to include considering issues related to common Node infrastructure and existing enterprise licenses, cost of porting to and re-certifying on a new platform, etc. (see the Migration Concern: Infrastructure Dependencies perspective).
- Assess the cost and benefits associated with conformance with relevant DoD initiatives.
- Compare actual versus estimated costs and benefits at each stage of the implementation.

P1232: Migration Concern: Risk Management

Migration to net-centricity presents risks. These include the standard risks associated with any system change as well as risks associated with increased dependencies outside of the scope of the program.

Program Actions:

- Characterize various migration options (see the Migration Patterns perspective) from the risk point of view.
- Include adequate reserve in the migration plans to accommodate anticipated risk.
- Clearly distinguish between program risk reduction activities (e.g., prototyping, analysis, experimentation) and product development activities. Carefully plan the transition from risk reduction activities to related product development activities. Include high-risk functionality in later increments to allow for early-on risk reduction activities.
- Identify and manage risks associated with anticipated reuse. Ensure that any intellectual property that is targeted for reuse (e.g., service descriptions, service implementations) is available, applicable, and affordable.
- Conduct component **Failure Impact Analysis**.
- Develop fielding procedures for software changes that ensure stability of the currently fielded functionality.
- Identify and manage risks associated with dependencies on common infrastructure (see the Migration Concern: Infrastructure Dependencies perspective).
- Identify and manage risks associated with dependencies with relevant DoD initiatives (see the Migration Concern: Conformance with Relevant DoD Initiatives perspective).

P1233: Migration Concern: Test in an Integrated Environment

Testing is a means to assess conformance with relevant DoD initiatives (see the Migration Concern: Conformance with Relevant DoD Initiatives perspective), assess conformance with net-centric technical tenets (see the Migration Concern: Conformance with Net-Centric Technical Tenets perspective), identify performance issues (see the Migration Concern: System Performance perspective), manage risks (see the Migration Concern: Risk Management perspective), and address a number of other key concerns. Migration to net-centricity and SOA, however, presents unique testing challenges - most noticeably in the use of a representative test environment.

For integrated testing, make sure to use integrated testing environments, including enterprise simulation platforms, where the mission flow could be tested for dependencies, as well **Service Level Agreements**, diagnostics instrumentation, etc.

Program Actions:

- Provide (or otherwise identify) an integrated test environment (e.g., enterprise simulation platforms, Service Level Agreement management capability, or diagnostics instrumentation; this environment might be external to the program.
- Test capabilities (to include the role of systems and services in operational processes) in an integrated test environment.
- Test service management mechanisms in an integrated test environment that reproduces or simulates the target operational environment.
- Prototype solutions and conduct performance tests.

P1234: Migration Concern: Migration Plan Maintenance

A good migration plan is critical to a successful net-centic migration. The migration plan must be kept current throughout the migration as requirements evolve, the enterprise or program architecture changes, and experience in executing the plan is gained. The longer-term phased migration plan spawns more detailed migration project plans for each phase of the migration.

Program Actions:

- Update migration plans at the end of each implementation increment.
- Update migration plans whenever a significant change in the enterprise or program architecture occurs.
- Update migration plans whenever the program becomes aware of a significant migration ***lesson learned***.
- Update related migration planning documentation whenever the migration plan changes (see the Plan Migration perspective).
- Update project plans for each increment of the migration just prior to executing that increment.

P1235: Migration Concern: Architecture Documentation Maintenance

The enterprise architecture documentation provides key enterprise-level operational, system, and technical contextual information for the migration. Similarly, the program architecture documentation provides key program-level operational, system, and technical contextual information. The program architecture must align to the enterprise architecture. Both the enterprise and the program architectures affect and are affected by the migration. As the migration progresses, update the documentation for the enterprise and the program architectures to keep the documentation current and aligned.

Program Actions:

- Track changes to the enterprise and program architectures during the execution of the migration plans.
- Coordinate changes to enterprise and program architectures documentation with key stakeholders.
- Update program architecture documentation to reflect changes made during the migration.
- Use the S300 Service Definition Framework (SDF) [R1216] to describe **services** across all architecture documents (also refer to *NESI Part 2: ASD(NII) Checklist Guidance* for additional SDF information.)

P1236: Migration Concern: Enterprise-Level Migration Knowledge Management

It is important to refer to the ***lessons learned*** from other migration efforts and to add to the store of migration lessons learned for others to use. Prepare migration documentation so that it might be shared outside of the program when circumstances permit.

Program Actions:

- Research prior lessons learned early in the migration planning process (see the Plan Migration perspective).
- Document migration lessons learned to include both successful and unsuccessful practices.
- Publicize the existence of documented migration lessons learned.
- Share migration plans and related documentation.
- Share architecture documentation to include XML data schemas, service descriptions, and descriptions of reusable architectural patterns.

P1203: Net-Centric and SOA Assessments

Assessment is a key tool in the migration to net-centricity and SOA. Assessment can be used to characterize the initial (***as-is***) state of the systems associated with a program, project the outcome of migration phases (***to-be***), and measure the actual progress of migration activities (and thus help update migration plans). Assessment helps to develop the rationale for the migration, develop and update migration plans, and report results.

Detailed Perspectives

- Assessment Considerations
- Phases of SOA Adoption

P1237: Assessment Considerations

There are a number of factors to consider when assessing net-centricity and SOA adoption:

- ***Distinction between "What" and "How Built" Assessments*** - Some assessments measure the specific types and numbers of networks and services that are built. Other assessments only measure the technical approach used to build those networks and services.

- ***Distinction between "Test" and "Assessment"*** - In a "test," the system is executed in a somewhat realistic environment to identify or otherwise explore performance or functionality-related issues. An "assessment" can also include visual inspections, design reviews, architecture reviews, etc., used to characterize the nature of a system in potentially many different dimensions.

- ***Program versus Enterprise-level Assessment*** - An individual program can be assessed to determine how well it meets its stated objectives - some of which may have an enterprise focus. A collection of program assessments, if performed using the same assessment methodology, can provide a picture of how that set of programs are achieving their objectives, some of which may be common. This is useful input into, but is not a replacement for, an overall assessment of how well the enterprise is achieving its objectives.

- ***Multiple Technical Dimensions*** - Net-centricity and SOA are complex technical subjects; therefore, a related assessment will have to address multiple technical dimensions (e.g., network standards, SOA standards, security standards, data distribution patterns). It is generally not useful to define net-centricity or SOA abstractly as a single characteristic (e.g., a "level" of net-centricity) or to identify one dimension as more important than another.

- ***Requirements and Funding Constraints*** - While theoretically it may be possible to define and achieve "perfect net-centricity" or "perfect SOA," most programs neither have the requirement nor the funding to achieve either goal. It is useful for assessments to identify clearly the as-is state, the target states for each Implementation increment, and the overall required and funded states as a percentage of the ideal state.

- ***Cost Versus Quality of Assessment*** - Balance the cost of performing the assessment against the quality of the assessment results and the anticipated benefits of the assessment. Some options include self-assessment supplemented by independent spot checks and partial assessments of high-priority or high-payback areas (e.g., key functional areas or operational processes).

- ***Net-Centric Assessment Versus SOA Assessment*** - A net-centric assessment (to include degree of connectivity, **Information Assurance** (IA), and non-SOA data sharing) may include a SOA assessment but may not provide a detailed characterization of the degree of adoption of SOA. An assessment focused on SOA may assume that connectivity and IA aspects are assessed elsewhere.

- ***Levels of Net-Centric Maturity*** - Many assessments provide discrete levels of net-centric maturity. While useful in some contexts, these levels may provide the generally false impressions that all levels must be traversed sequentially (i.e., that all parts of a program must migrate from one level to the next in lock-step) and that all programs must achieve some minimum level at some point in time. Instead, metrics that depict a multi-dimensional view of the state of net-centricity are more useful. Use the ASD(NII) Net-Centric Checklist [R1177] to derive such metrics.

- ***Levels of SOA Maturity*** - These are typically attempts to formalize the gradual evolution of SOA adoption across the enterprise in order to identify the appropriate multi-staged governance actions. See the Phases of SOA Adoption perspective for a high level and approximate characterization of a time-phased adoption of SOA.

Best Practices

- BP1841: Involve key stakeholders in the assessment of the as-is architecture in preparation for the migration to net-centricity and SOA.

- BP1847: Use the same assessment methodology to assess the as-is architecture, define the target (to-be) architecture for each migration increment, and assess migration progress at the end of each migration increment.

P1238: Phases of SOA Adoption

An analysis of current industry practice reveals general phases, or levels, of SOA adoption. The following "maturity phases" are adopted primarily from 2003 CBDI industry forum reports [R1208] :

1. Early Learning
2. Integration
3. Reengineering
4. Maturity

This categorization is not absolute and should not be perceived as prescriptive. Rather, it provides a useful basis for assessing the current or projected (***to-be***) state of SOA adoption in a program or enterprise and for supporting migration planning. More specifically, it can help to identify the amount of development effort to apply to achieve an intended pedigree of a service [R1216] . As discussed below, the pedigree of a service can change as the state of SOA adoption evolves.

Early Learning

This phase is mostly exploratory and involves the tactical implementation of services. "Moving beyond this level requires a change in business practices, not just technology." [R1208]

At this level of SOA adoption, enterprise communities often begin to analyze existing architectures to identify common approaches and places to exploit net-centricity and SOA. The view is towards strategic requirements, cross-organizational standardization, and collaborative system planning.

Drivers:

- Better application integration (e.g., **Web services** are simply viewed as a better form of middleware that allows platform neutral information sharing)
- Short term, low risk return on investment (ROI)
- Experimentation
- Early successes in delivery mission agility through SOA

Service Implementations:

- Focused on information access
- Harvest existing implementations/interfaces as **services** (some applications are simply wrapped to establish some initial SOA)
- Technical service (one that supports the implementation and is not meant to be shared) rather than mission/business service
- Specific function
- Services are not yet linked into an operational process
- Standalone projects
- Focus on internal (and some low risk external) services implementations

- Service delivery and support managed under existing processes
- Service consumers known in advance (no need for run-time discovery mechanisms such as **UDDI**)
- Use existing security mechanisms

Integration

In the Integration phase, the organization builds on successes of the Early Learning phase of SOA adoption; "the organization is likely to see opportunities from integration with existing core systems." [R1208]

Drivers:

- Focus on business/mission needs (users begin to see the benefits)
- SOA becomes a strategic objective
- Elimination of gaps and redundancies
- Reuse

Service Implementations:

- Operational process oriented
- Some business process integration (e.g., with the use of **BPEL**)
- Architected (rather than opportunistic)
- Separation of provider and consumer applications
- Considerations given to support of shared services (basic **Service Level Agreement** and management capabilities might be established)
- Mostly internal usage
- Service delivery and support use extensions of existing processes (some delivery process changes)
- Service consumers known in advance (no need for run-time discovery mechanisms such as UDDI)
- Use more sophisticated service security mechanisms

Reengineering

In the Reengineering phase, the service becomes a business product. "The very purpose of services will undergo change, as well as the delivery technology and practices." [R1208]

In most cases, the service itself would be reengineered from a standalone local source of information into a product-like unit of a capability that is reliable, adaptable, potentially composeable, secure, and scalable. Business models are reengineered as well to take advantage of the SOA paradigm. Note that not every service will be reengineered.

Drivers:

- Availability of both basic business and technical services (those that support the implementation and are not meant to be shared)
- Availability of the service infrastructure

- Guarantees of availability, reliability and performance
- Potentially newer, simpler, standards based technology

Service Implementations:

- Enterprise level
- Services used across organizations (pedigree of some services changes to support broader requirements)
- Services implemented as an integral part of reliable capabilities
- Services used as black box components where provider and consumer processes are separate
- Automated **discovery**
- Supported by mature security standards
- Comprehensive Service Level Agreements between providers and consumers
- Standard-based monitoring and governance

Maturity

In the Maturity phase, most of the primary capabilities and processes of the organization are aligned with the SOA concept.

Drivers:

- Seamless enterprise level integration
- Reuse of services

Service Implementations:

- Services are ubiquitous
- Federated
- Complex products, orchestrated from potentially different providers and cross organizations
- Machine-readable contracts
- On-demand execution
- Some of the services become mandated standard
- Monitored and controlled at a business level of abstraction

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.

Guidance and Best Practice Details

BP1835

Statement:

Develop a formal migration plan to support the migration to **net-centricity** and SOA.

Rationale:

Most **net-centric** and SOA migrations are expected to be lengthy and subject to many influencing and changing factors. As a result, they should be documented in an organized manner. The migration plan will then be available to guide migration activities. Even small-scale migrations will benefit from having a formal migration plan, but the migration plan will be correspondingly less complex and easier to generate and maintain.

Justifies:

BP1836 BP1837 BP1842 BP1844

Referenced By:

Migration Planning Process
Finalize Migration Plan

Evaluation Criteria:

1) Test: [BP1835.1]

Does the project have a formal migration plan to support migration to net-centricity and SOA?

Procedure:

Verify the presence of a formal migration plan supporting migration to net-centricity and SOA.

Example:

BP1836

Statement:

Obtain consensus on the migration plan from all key **stakeholders**.

Rationale:

The stakeholders present varying viewpoints about issues associated with the migration plan. Obtaining consensus from key stakeholders on the migration plan can prevent critical miscommunication and support the management of expectations.

Derived From:

BP1835

Referenced By:

Migration Planning Process
Finalize Migration Plan

Evaluation Criteria:

1) Test: [BP1836.1]

Does the migration plan identify key stakeholders?

Procedure:

Examine the migration plan and verify that it identifies key stakeholders.

Example:

2) Test: [BP1836.2]

Does the migration plan reflect key stakeholders' involvement and input?

Procedure:

Examine and analyze the migration plan to confirm that it reflects key stakeholders' involvement and input.

Example:

BP1837

Statement:

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.

Derived From:

BP1835

Referenced By:

Migration Planning Process
Finalize Migration Plan

Evaluation Criteria:

1) Test: [BP1837.1]

Does the migration plan track its currency date and any updates?

Procedure:

Examine the migration plan for a currency date and update tracking.

Example:

BP1838

Statement:

Develop as-is architecture artifacts to support the migration to **net-centricity** and SOA.

Rationale:

CJCSI 6212.01D requires the following architecture products: AV-1, OV-1 (optional), OV-5, OV-6c (optional), SV-1 (optional), SV-5, SV-6, and TV-1. **DoDAF** V1.5 describes each of these products.

Referenced By:

Prepare for Migration

Evaluation Criteria:

1) Test: [BP1838.1]

Does the program have the required architecture products describing the as-is architecture?

Procedure:

Verify the existence of architecture products describing the as-is architecture.

Example:

2) Test: [BP1838.2]

Do the program as-is architecture products support the migration planning process?

Procedure:

Assess the quality (e.g., breadth, depth, correctness) of the as-is architecture products.

Example:

BP1839

Statement:

Perform a business case analysis to support the migration to net-centricity and SOA.

Rationale:

Analyzing the migration as a business case establishes the connection between the prospective technical improvements and their purpose, provides clarity about expected benefits and costs, and sets the stage for migration planning and evaluation of the achieved results during the execution.

Referenced By:

Assess Migration Needs

Evaluation Criteria:

1) Test: [BP1839.1]

Does the program have a documented business case analysis in support of net-centric and SOA migration?

Procedure:

Verify the existence of a documented business case analysis that supports net-centric and SOA migration.

Example:

BP1840

Statement:

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.

Justifies:

BP1846

Referenced By:

Assess As-Is Requirements

Evaluation Criteria:

1) Test: [BP1840.2]

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:

2) Test: [BP1840.1]

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:

BP1841

Statement:

Involve key stakeholders in the assessment of the as-is architecture in preparation for the migration to net-centricity and SOA.

Rationale:

The stakeholders present varying viewpoints about issues associated with the as-is architecture. Involving them early on in the migration planning process provides key input, "sanity check," and potential advocacy. Achieve consensus on the assessment among key stakeholders.

Referenced By:

Assess As-Is Architecture
Assessment Considerations

Evaluation Criteria:

1) Test: [BP1841.1]

Does the as-is architecture analysis document reflect the involvement of key stakeholders in its preparation?

Procedure:

Verify that as-is architecture analysis document pertaining to net-centric and SOA migration reflects the involvement of key stakeholders.

Example:

BP1842

Statement:

Formally document the migration rationale to support the migration to net-centricity and SOA.

Rationale:

A clearly documented rationale presents the business case for the migration to all stakeholders.

Derived From:

BP1835

Justifies:

BP1843

Referenced By:

Develop Migration Rationale Statement

Evaluation Criteria:

1) Test: [BP1842.1]

Does the program have a migration rationale statement to support the migration to net-centricity and SOA?

Procedure:

Review migration planning documents to verify they include a migration rationale statement.

Example:

2) Test: [BP1842.2]

Does the Migration Plan include a formally documented migration rationale?

Procedure:

Review the Migration Plan to verify it includes a migration rationale.

Example:

BP1843

Statement:

Obtain consensus among all key stakeholders on the rationale for the migration to net-centricity and SOA.

Rationale:

The stakeholders present varying viewpoints about issues associated with the migration. Involving them early on in the migration planning process provides key input and potential advocacy.

Derived From:

BP1842

Referenced By:

Develop Migration Rationale Statement

Evaluation Criteria:

1) Test: [BP1843.1]

Does the Migration Rationale statement explicitly demonstrate the consensus on the rationale for the migration to net-centricity and SOA among all of the key stakeholders?

Procedure:

Review the Migration Rationale statement and verify that it demonstrates all key stakeholders consensus.

Example:

BP1844

Statement:

Develop a vision statement for the migration to net-centricity and SOA.

Rationale:

A vision statement provides strategic direction for the migration. It describes the high-level, time-indeterminate state of the target of the migration. The vision statement will be documented in the migration plan.

The vision for the program indicates the desired long-term direction for the system. It offers a view of its evolution and, potentially, eventual replacement. The vision for the program shows the scope of the system within its larger context (the enterprise); thus, the vision for the program should be consistent with the higher headquarters vision statements. Similarly, the vision for the migration to net-centricity and SOA should be consistent with the vision for the program.

Derived From:

BP1835

Referenced By:

Develop Alternative Target Architectures

Evaluation Criteria:

1) Test: [BP1844.1]

Does the migration plan contain a vision statement for the migration?

Procedure:

Review the migration plan and verify that it contains a migration vision statement.

Example:

BP1845

Statement:

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:

Develop Implementation Plans
Critical Migration Concerns

Evaluation Criteria:

1) Test: [BP1845.1]

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:

BP1846

Statement:

Involve key stakeholders in the development of the implementation plan increments.

Rationale:

The stakeholders present varying viewpoints about issues associated with the migration. Involving them in the migration planning process provides key input and potential advocacy.

Derived From:

BP1840

Referenced By:

Develop Implementation Plans

Evaluation Criteria:

1) Test: [BP1846.1]

Does the implementation plan for net-centricity and SOA migration contain considerations of key stakeholders?

Procedure:

Review the migration plan tasks and verify that they address key stakeholders' concerns.

Example:

BP1847

Statement:

Use the same assessment methodology to assess the as-is architecture, define the target (to-be) architecture for each migration increment, and assess migration progress at the end of each migration increment.

Rationale:

Using the same assessment methodology provides a consistent basis for the comparison of results. NESI can serve as a source for an assessment mechanism. For example, NESI Part 2 provides specific detailed guidance for addressing the ASD(NII) Net-Centric Checklist [R1177] requirement.

Referenced By:

Assess As-Is Architecture
Migration Concern: Conformance with Net-Centric Technical Tenets
Assessment Considerations

BP1848

Statement:

Develop one or more target architectures for the migration.

Rationale:

The target architectures depict potential migration solutions. Multiple alternative target architectures model differences in cost and performance that can lead to the selection of the migration target. The target architecture can also help to identify operational, system, and technical issues that may affect the migration.

Referenced By:

Develop Alternative Target Architectures

BP1849

Statement:

Delay the decoupling of interface from implementation until the migration to a standard interface.

Rationale:

It may not make sense to decouple a non-standard interface from its implementation if a plan exists to move to a standard interface in the near future (and perform the de-coupling then).

Referenced By:

Pattern: Exposing Functionality through Non-standard Interfaces

BP1850

Statement:

Use service design guidelines and best practices to convert an interface into a service.

Rationale:

A consistent approach to service development aids in interoperability and provides efficiencies in the use of development resources. When converting an interface into a service, refer to the service design best practices in NESI Part 2. As a service is a type of a public interface, refer to NESI Part 5 Public Interface Design perspective [P1060].

Referenced By:

Pattern: Wrapping Legacy Code into a Service
Pattern: Exposing Web Services

BP1851

Statement:

Focus wrapping efforts on key operational processes.

Rationale:

A focus on key operational processes helps to direct the attention of service development activities on areas of high return on investment. In addition, there are COTS tools available that wrap legacy applications into WSDL-based services. This Wrapper code typically runs on an Application Server. See DoD 8320.02-G [R1217] for a discussion of how to assess potential candidates for wrapping

Referenced By:

Pattern: Wrapping Legacy Code into a Service
Pattern: Exposing Web Services

BP1852

Statement:

Align software components with operational processes.

Rationale:

Aligning software components with operational processes most effectively prepares the system for further SOA implementation.

Referenced By:

Pattern: Re-Implementation

BP1853

Statement:

Apply applicable service development best practices to software component development.

Rationale:

Applying applicable service development best practices to software components (e.g., developing components for flexibility and reuse) - even before they get exposed as services – can provide desirable architectural qualities and lay the foundation for future service implementation.

Referenced By:

Pattern: Re-Implementation

BP1854

Statement:

Use **SOAP** over **JMS** only when implementation interoperability is not the main driving factor.

Rationale:

The underlying implementations of the interface are vendor-specific and, therefore, require bridging so that two JMS communicating peers from different vendors can interoperate. There is no interoperability standard yet available for SOAP over JMS.

Referenced By:

Pattern: SOAP over JMS

BP1855

Statement:

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:

Net-Centric Data Strategy (NCDS)

BP1856

Statement:

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:

Net-Centric Data Strategy (NCDS)

BP1857

Statement:

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:

Net-Centric Data Strategy (NCDS)

BP1858

Statement:

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:

Net-Centric Data Strategy (NCDS)

BP1859

Statement:

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:

Net-Centric Data Strategy (NCDS)

BP1860

Statement:

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:

Net-Centric Data Strategy (NCDS)

BP1861

Statement:

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:

Net-Centric Data Strategy (NCDS)

Glossary

.NET		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, 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.
Acquisition Program Baseline	APB	Each program's APB is developed and updated by the program manager and will govern the activity by prescribing the cost, schedule and performance constraints in the phase succeeding the milestone for which it was developed. The APB captures the user capability needs, including key performance parameters, which are copied verbatim from the capability development document. (Source: CJCSI 3170.01E, *Joint Capabilities Integration and Development System*, 11 May 2005, Glossary page GL-4)
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.
Assistant Secretary of Defense for Networks and Information Integration	ASD (NII)	(Source: http://www.dod.mil/nii/)
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)
Capability		Capability is the ability to execute a specified course of action. A capability may or may not be accompanied by an intention. Capabilities include personnel, processes, and materiel and are represented using architecture products. Source: JP 1-02, 12 Apr 2001 *"Department of Defense Dictionary of Military and Associated Terms"*
Chairman of the Joint Chiefs of Staff	CJCS	The Goldwater-Nichols DOD Reorganization Act of 1986 identifies the Chairman of the Joint Chiefs of Staff as the senior ranking member of the Armed Forces. As such, the Chairman of the Joint Chiefs of Staff is the principal military

		adviser to the President. (Source: http://www.jcs.mil/chairman/chairman_resp.html)
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)
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 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.
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 nd 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*)
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**.*
Data		Unprocessed information; information without context.
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)
Discovery		Search, locate or publish data (content), other capabilities (services), or users across the **Global Information Grid** (GIG).
Doctrine, Organization, Training, Materiel, Leadership, Personnel, Facilities	DOTMLPF	

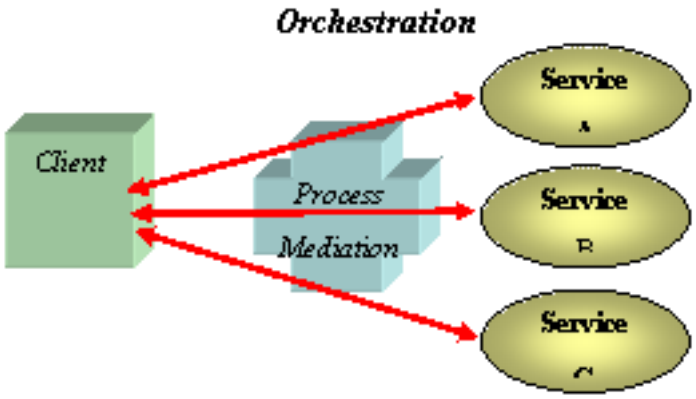
Document Type Definition	DTD	An optional part of the XML document prolog, as specified by the XML standard. The DTD specifies constraints on the tags and tag sequences that can be in the document. The DTD has a number of shortcomings, however, and this has led to various schema proposals. (Source: http://java.sun.com/j2ee/1.4/docs/glossary.html)
DoD Architecture Framework	DoDAF	Defines a common approach for DoD architecture description, development, presentation, and integration for both warfighting operations and business processes [DoDAF v1.0 supersedes C4ISR Architecture Framework v2.0, 18 December 1997]. (Source: Office of the Secretary of Defense memo of 9 Feb 2004, *The Department of Defense Architecture Framework (DoDAF)*)
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)
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 Service Bus	ESB	A layer of middleware through which a core set of reusable business services are made available.
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)
Failure Impact Analysis		Failure Impact Analysis is a process of analyzing a particular hardware/software configuration to determine the true impact of any individual failed component. Source: http://wiki.ittoolbox.com/index.php/Component_Failure_Impact_Analysis
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)
General Public License	GPL	A license that defines a specific set of distribution terms for free software. A GPL specifically does not let redistributors add any additional restrictions when they redistribute or modify the software. This means that every copy of the software, even if it has been modified, must be free software. (Source: http://www.gnu.org/copyleft/gpl.html)
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.
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)
Information Assurance	IA	Measures taken to protect and defend our information and information systems to ensure Confidentiality, Integrity, Availability, and Accountability, extended to restoration with protect, detect, monitor, and react capabilities.
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 6212.01D, 8 March 2006, Glossary page GL-11)
Interface		The functional and physical characteristics required to exist at a common boundary or connection between systems or items. (Source: DoD 4120.214-M)
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 x 10^9) addresses. IPv6 supports up to about 3.4 x 10^{38} (340 undecillion) addresses. This is the equivalent of 4.3 x 10^{20} (430 quintillion) addresses per square inch (6.7 x 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)
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 Message Service	JMS	An API for invoking operations on enterprise messaging systems. (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/)
JavaServer 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 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)
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)
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 • Aggregation • Adaptation • Orchestration • Choreography
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.
Multi-Purpose Internet Mail Extensions	MIME	
Net-Centric		Exploitation of advancing technology that moves from an application centric to a data-centric paradigm - that is, providing users the ability to access applications and services through Web services - an information environment comprised of interoperable computing and communication components. (Source: ASD(NII) *Net-Centric Checklist* v2.1.3, 12 May 2004)
Net-Centricity		Net-centricity 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 self-synchronization. In essence, net-centricity translates information superiority into combat power by effectively linking knowledgeable entities in the battlespace. (Source: ASD(NII) Net-Centric Checklist v2.1.3, 12 May 2004)
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)
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)  ***Note:*** *See* ***Mediation****.*
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.
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)
Service		A service is any function that has a clearly defined interface accessed through well-defined public access points.
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.
Simple Mail Transfer Protocol	SMTP	
Simple Object Access Protocol	SOAP	SOAP is a lightweight XML-based messaging protocol used to encode the information in **Web service** request-and-response messages before sending them over a network. SOAP messages are independent of any operating system or protocol and may be transported using a variety of Internet

		protocols, including **SMTP**, **MIME**, and **HTTP**. (Source: http://www.webopedia.com/TERM/S/SOAP.html)
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
Stakeholder		An enterprise, organization, or individual having an interest or a stake in the outcome of the engineering of a system. (Source: EIA-632, Annex A)
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 Development Life Cycle	SDLC	The Systems Development Life Cycle is a process of information systems development that encompasses a systematic approach for all its phases, including analysis, design, implementation and maintenance. Source: http://en.wikipedia.org/wiki/Systems_Development_Life_Cycle
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)
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)

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-enable		Web-enable is the process of make existing computer applications available to users from a standard Web browser. Source: http://www.bitpipe.com/tlist/Web-Enablement.html
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	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.
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)

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)

References

R1164	DoD Directive 5000.1, *The Defense Acquisition System*, 12 May 2003 (certified current as of 24 November 2003); http://www.dtic.mil/whs/directives/corres/pdf/500001p.pdf.
R1165	DoD Instruction 5000.2, *Operation of the Defense Acquisition System*, 12 May 2003; http://www.dtic.mil/whs/directives/corres/pdf/500002p.pdf.
R1166	DoD Directive 8100.1, *Global Information Grid (GIG) Overarching Policy*, 19 September 2002 (certified current as of 21 November 2003); http://www.dtic.mil/whs/directives/corres/pdf/810001p.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 Architecture Framework (DoDAF)*, Version 1.5, 23 April 2007; https://dars1.army.mil/IER/index.jsp
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 3170.01F, *Joint Capabilities Integration and Development System*, 01 May 2007; http://www.dtic.mil/cjcs_directives/cdata/unlimit/3170_01new.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 6212.01D, *Interoperability and Supportability of Information Technology and National Security Systems*, 8 March 2006; 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.
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
R1208	CBDI 2003 report *"Web Services Roadmap"*; http://roadmap.cbdiforum.com
R1209	AFEI report *"Facilitating Shared-Services in the DoD"*; Feb 12, 2006; http://www.afei.org/news/documents/DS3SharedServicesPaperFinalVersion021206c.pdf
R1210	Randy Heffner, Forrester Research; Executive Summary from *"Your Path to SOA"*; December 7, 2004; http://www.forrester.com/go?docid=34813
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
R1212	Jason Bloomberg & Ron Schmelzer, ZapThink, LLC *"SOA Pilots"* (presentation as part of a full-day seminar for Tarantell in Oslo, Norway on November 16, 2006); http://www.zapthink.com/report.html?id=ZTP-0277 [user access required]
R1213	David Margulius, *"Banking on SOA"*; InfoWorld; July 13, 2006; http://www.infoworld.com/article/06/07/13/29FEwachovia_1.html
R1214	H.M. Sneed, *"Wrapping Legacy Software for Reuse in SOA"*; Dec 2005
R1215	Yefim V. Natis, Gartner; *"Service-Oriented Architecture Scenario"*; 16 April 2003; http://www.gartner.com/DisplayDocument?doc_cd=114358
R1216	Global Information Grid Net-Centric Implementation Document - Service Definition Framework (S300); 21 December 2005 (V2.0)
R1217	DoD 8320.02-G, April 12, 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)