

UNCLASSIFIED



Net-Enabled Command Capability (NECC)

DEVELOPER'S HANDBOOK

23 April 2008

Version 1.4.0

NECC-DHB-00033

Prepared by:

Net-Enabled Command Capability
Joint Program Management Office (JPMO)
P.O. Box 4502
Arlington, VA 22204-4502

In Collaboration with the NECC Component Program Managers for Navy, Air Force, Army, Marine Corps, USJFCOM, and DISA

DISTRIBUTION – Distribution authorized to DoD and DoD Contractors only; for administrative/operational use (April 2008). Other requests for this document shall be referred to the NECC Program Management Office.

DESTRUCTION NOTICE – For unclassified, limited documents, destroy by any method that will prevent disclosure of contents or reconstruction of the document.

UNCLASSIFIED

APPROVAL PAGE

Submitted by:



Date:

7/28/08

MARK KUZMA

Chief Engineer,
Net-Enabled Command Capability (NECC),
Defense Information Systems Agency (DISA)

REVISION HISTORY

REVISION NUMBER	REVIEWER / ORG	CHANGES	REVISION DATE	NAME OF PERSON ENTERING CHANGE
0.8	NECC JPMO Systems Engineering (SE) Branch	Reorganized and updated the document based on v.0.7 to become consistent with Increment 1 architecture documents, Systems Engineering Plan, and other documents.	20 Jul 2007	20 Jul 2007
0.9	NECC JPMO SE Branch	Incorporated comments from internal review of v.0.8. Added contents on training materials. Added document templates. Added user interface samples in 0.	31 Aug 2007	31 Aug 2007
0.9.1	NECC JPMO SE Branch	Incorporated additional comments from internal review.	07 Sep 2007	07 Sep 2007
0.9.2	NECC JPMO SE Branch	Adjudicated review comments from v.0.9.1. Revised based on the comments.	12 Oct 2007	10 Oct 2007
1.0	NECC JPMO SE Branch	Finalized responses to comments based on guidance from Architecture Working Group. Also incorporated internal review comments from v.0.9.2.	02 Nov 2007	02 Nov 2007
1.1	NECC JPMO SE Branch	Changed naming convention to include materiel solution after the CM name	12 Dec 2007	12 Dec 2007
1.2	NECC JPMO SE Branch	Updated links to DKO	24 Jan 2008	24 Jan 2008
1.3	NECC JPMO SE Branch	Incorporated Change Request NEC00011 (versioning), updated contents from the Breadcrumbs Guide, updated NCES and MDR information, synchronization with technical standards, and reference to Request for Deviation procedure.	17 Mar 2008	17 Mar 2008
1.4	NECC JPMO SE Branch	Updated information for SE reviews. Incorporated and synchronized with deliverable list from Developmental Work Package template (see 0). Added 0 with CM deployment information and checklist at GCN. Updated references to templates and SOPs.	18 Apr 2008	18 Apr 2008
1.4	JPMO/Program Control	Edited document. Fixed all references and hyperlinks to DKO. Correct all first time definitions of acronyms. Reformatted document in new NECC template. Fixed internal hyperlinks. Verified all references, hyperlinks, and section links. Add and deleted acronyms to Appendix J. Corrected Appendix F title. Reset Table definitions. Updated TOC.	22 Apr 2008	23 Apr 2008

TABLE OF CONTENTS

APPROVAL PAGE.....II

REVISION HISTORY III

1 INTRODUCTION..... 1

2 SYSTEMS ENGINEERING FOR CAPABILITY MODULES..... 1

2.1 DESIGN AND DEVELOPMENT STEPS 1

2.2 ARTIFACTS FOR CAPABILITY MODULES 2

2.3 TEST AND EVALUATION 3

2.4 FEDERATED DEVELOPMENT AND CERTIFICATION ENVIRONMENT AND RELATED REGISTRIES 4

2.5 CONFIGURATION MANAGEMENT 6

2.6 CAPABILITY PACKAGE DELIVERY PROCESS 8

2.7 CERTIFICATION, ACCREDITATION, AND INFORMATION ASSURANCE 9

2.8 NET-CENTRIC DATA STRATEGY COMPLIANCE 11

2.9 OPERATION AND SUPPORT 12

2.10 TRAINING MATERIALS 12

3 DESIGN AND DEVELOPMENT GUIDANCE..... 13

3.1 SERVER ENVIRONMENT 13

3.2 SCHEMA DEVELOPMENT 14

3.3 DATA SERVICE DEVELOPMENT 15

3.4 HUMAN COMPUTER INTERFACE AND ASYNCHRONOUS JAVASCRIPT AND XML (AJAX) 16

3.5 NET-CENTRIC ENTERPRISE SERVICES 18

3.6 MIGRATING TO THE NECC SOA 21

APPENDIX A – REFERENCES 22

APPENDIX B – CROSSWALK WITH INCREMENT 1 ARCHITECTURE..... 25

APPENDIX C – SYSTEM ENGINEERING REVIEWS AND DELIVERABLES 26

APPENDIX D – INFORMATION FOR PROVISIONING CMS AT GCNS 32

APPENDIX E – USAGE OF STANDARDS FOR NECC CAPABILITY MODULES..... 34

APPENDIX F – XML SCHEMA EXAMPLE..... 35

APPENDIX G – AJAX APPLICATION EXAMPLES 37

APPENDIX H – SCORM COMPLIANT COURSEWARE 43

APPENDIX I – CONTACT INFORMATION..... 44

APPENDIX J – ACRONYMS 45

LIST OF FIGURES

Figure 1: CM Information and Links in FDCE and Other Related Registries 5

Figure 2: Nominal CM Development Milestones and Deliverables 26

Figure 3: A Graphic View of the TargetContext Schema 35

Figure 4: A Multi-pane View with a Form and a List..... 38

Figure 5: A Multiple Pane View with Groups of Objects in the List..... 39

Figure 6: An AJAX-enabled Map View with a list pane on the left (from Google)..... 40
Figure 7: An AJAX-enabled Map View with user selected tools (from Google)..... 41
Figure 8: An AJAX-enabled Tree and Chart View..... 42

LIST OF TABLES

Table 1: CM Artifacts Provided to Developers 2
Table 2: Sites and Links of FDCE and Other Related Registries 6
Table 3: Capability Package Examples..... 8
Table 4: Guidelines for Net-Centric Data Strategy Compliance 11
Table 5: Metrics Types for Services 12
Table 6: Approved Operating Systems for NECC..... 14
Table 7: Virtual Machine Configuration Parameters..... 14
Table 8: Core Enterprise Services provided by NCES and their usages by NECC CMs 18
Table 9: Reference Documents for the Handbook..... 22
Table 10: Crosswalk with Increment 1 Architecture Documents 25
Table 11: Artifacts for Systems Engineering Reviews 26
Table 12: NECC Design Review Checklist..... 27
Table 13: Deliverables from Developers 28
Table 14: Information for Provisioning CMs at GCNs..... 32
Table 15: Usage of Standards for NECC CMs 34
Table 16: A Partial View of the TargetContext Schema..... 35
Table 17: AJAX Application Examples..... 37
Table 18: NECC Contact Information for Developers 44

1 INTRODUCTION

This Handbook is intended for materiel provider staff and developers of Net-Enabled Command Capability (NECC) Capability Modules (CMs). Following the receipt of a CM Work Package, NECC developers will be engaged in software design, development, integration, testing, and other systems engineering activities. This handbook introduces the key engineering concepts used in NECC and serves as a practical guide for NECC developers.

For each topic covered below, a summary is provided based on other NECC documents (such as the Capability Development Document (CDD) [Ref. 1], Increment 1 architecture documents [Ref. 2] and Systems Engineering Plan (SEP) [Ref. 3]). Requirements for developers are derived from these documents.

The keyword "MUST" or "SHALL" in this handbook indicates that developers must follow a stated instruction, whereas "SHOULD" means that due diligence must be performed before deviating from a stated instruction. Unless otherwise specified, developers in this document refer to NECC developers.

When appropriate, discussions on specific guidance, conventions, best practices, and examples are presented either in the main text or in the appendices. The reference section provides full links to the relevant documents.

This handbook, along with the NECC Systems Engineering (SE) processes, helps improve productivity by lowering the learning curve for new developers and ensures the consistency of NECC software products.

2 SYSTEMS ENGINEERING FOR CAPABILITY MODULES

This section introduces the NECC SE processes related to the development and certification of CMs.

2.1 Design and Development Steps

The NECC SEP [Ref. 3] describes the requirement analysis, design, and build process for CMs. A Capability Definition Package (CDP) is decomposed into one or more Engineered Mission Threads (EMTs). An EMT is a use case that details the user's process and activities. EMTs are among the items used to develop Test, Evaluation, and Certification (TEC) Criteria and Test and Evaluation (T&E) scenarios (also known as Operational Mission Threads). Following the analysis of these EMTs, functional and performance requirements are allocated to CMs, which are further defined by the CM Specifications in work packages.

The developmental work packages describe the work to be performed and products to be developed by developers. The following is an overview of activities that involve developers (the activities map to those in Steps 7 and 8 of the SEP):

- Design – MUST be based on the requirements in the work packages. CM Preliminary Design Review (PDR), Information Assurance Readiness Review (IARR), and Critical Design Review (CDR) SHALL be conducted. See Appendix C for checklists related to these reviews.
- Registration of CM – developers MUST register the CM at the Federated Development and Certification Environment (FDCE) before the CDR. The CM enters the Development stage

at this point. Ref. 10 gives step-by-step instructions on registering CMs at the FDCE. As part of this process, developers **MUST** register CM metadata products with the Department of Defense (DoD) Metadata Registry [Ref. 5] and service information with the Service Registry of Net-Centric Enterprise Services (NCES) [Ref. 6].

- Development and Integration – develop software that implements the services under a CM. Each software module **MUST** include automated unit testing.
- Testing and Evaluation – **MUST** be based on the TEC Criteria [Ref. 7]. The CM **SHALL** only transition to the Developmental Piloting stage and the subsequent Operational Piloting stage upon successful completion of Test and Evaluation (T&E).

During the Development and Developmental Piloting stages, Component Program Management Offices (CPMOs) and developers perform the bulk of the CM testing and evaluation and prepare test reports. Upon approval, a CM is transitioned to the next stage - Operational Piloting. Details about the stages of CM development are in the Systems Engineering Plan [Ref. 3] and FDCE Business Plan and Concept of Operations [Ref. 9].

2.2 Artifacts for Capability Modules

Table 1 lists the CM artifacts provided to developers by NECC Joint Program Management Office (JPMO) as part of a CM developmental work package. The information in a specific CM Work Package may supersede this handbook.

Table 1: CM Artifacts Provided to Developers

#	ARTIFACTS	DESCRIPTIONS
1	Statement of Work	Describes the scope of the work and deliverables. It also includes internal and external systems engineering and configuration management requirements, the required level of support for internal and external piloting and certification activities, including Capability Provisioning Events (CPEs), Operational Tests [Ref. 8], and Operational Concept Experiments/Events (OCEs) [Ref. 9].
2	CM Specification	Specifies a CM's functional and performance requirements. It contains interface, physical, software, data, security, technical operations, and integrated logistics support requirements, constraints and limitations on materiel solutions, and Joint/Service implementation requirements.
3	Test, Evaluation, and Certification Criteria	Defines the set of standards and measures that must be tested in order to validate the requirements for the CM.
4	Execution Requirements	Describes the cost, schedule, and performance reporting for a CM. Describes the coordination of developers with the JPMO, Joint Forces Command (JFCOM), Joint System Team (JST), and CPMOs. It also includes technical governance, issue resolution process, and JPMO review schedule.

The CM Specification is derived from a Reference Component Model, which is an intermediate design artifact that portrays candidate CMs and their exposed service interfaces. If a proposed CM solution does not completely satisfy the CM Specification (called Service Performance Specification (SPS) in Ref. 3), an evaluation and reconciliation will be conducted as described in

section 3.1.4 of the SEP [Ref. 3]. Any variance between the CM Specification and the CM to be delivered MUST be captured by a Request for Deviation (RFD), which needs to be approved by the NECC Configuration Control Board (CCB) or the Configuration Management Board (CMB) before the CDR. Developers MUST follow the RFD procedure [Ref. 4], which also provides a RFD form.

Developers may also be involved in issue resolution. The issue resolution process is defined in Memorandums of Agreement (MOAs) that NECC will establish with other programs/organizations (cf. section 4.5.4 of the SEP). The JPMO, Joint Combat Capability Developer (JCCD), and relevant CPMOs will jointly monitor the process.

The CM Developmental Work Package lists the contractual deliverables for a specific CM. A CM may be packaged into one or more Capability Packages for deployment. Capability Packages are standardized software packages for deployment at Global Information Grid (GIG) Computing Nodes (GCN). In addition to software developed for NECC, a Capability Package contains Commercial Off-The-Shelf (COTS) software, Government Off-The-Shelf (GOTS) software, and Guest Operating System (OS) to be hosted in a virtual machine (see Section 3.1 for details).

Table 13 in Appendix C lists the artifacts to be delivered by developers. In the Developmental Work Package, the deliverable list for a specific CM may be tailored from this table. Note that the Software End Items include automated test drivers, which will be used to validate the installation and configuration of a CM based on the test procedure. The Software Design Description (SDD) and Interface Design Description (IDD) are reviewed in design reviews. Appendix C also gives checklists for systems engineering reviews. Developers SHOULD consult the checklists when preparing for the reviews.

2.3 Test and Evaluation

As described in the NECC Test and Evaluation Master Plan (TEMP) [Ref. 8], the NECC program uses a two-tiered integrated test organizational approach. The JST is the first level of the test hierarchy. It is an O-5/O-6 working level organization with representation from all test and certification stakeholders, including the Lead Operational Test Agency (Army Test and Evaluation Command (ATEC)), JPMO, CPMOs, Joint Interoperability Test Command (JITC), and JCCD.

The second level of the test hierarchy is the CM Test Team (CMTT), composed of an Operational Test Lead and a CPMO Development Test Lead. The Operational Test Lead is assigned by the lead Operational Test Agency based on recommendations from the JST. It is the responsibility of the CMTT to assess a specific CM or a combination of CMs based on a mission thread in the TEMP Annexes. The CMTT works closely with the Integration and Technology Piloting (I&TP) team, the Information Assurance (IA) Validator, and the JST to ensure synergy and efficiency.

NECC uses the TEC Criteria [Ref. 7] to define the set of tests that are needed to validate the requirements. These tests are divided into groups based on the source of the requirements, including Mission Capability, Performance, or Net-Ready Key Performance Parameters (NR-KPP). The requirements are also assigned to the appropriate stages of the FDCE process. For each CM, the JST will review and tailor the TEC Criteria. Details of this process are in the TEMP [Ref. 8]. Developers MUST participate in the tailoring and assertion process of the

tailored TEC Criteria. Developers **MUST** also develop the Test Procedure for a CM (see Test Procedure and Report in Table 13, number 5) in collaboration with the JST based on the tailored TEC Criteria.

The three stages of CM development progressively demonstrate that the tailored TEC Criteria are satisfied. The Development stage focuses on capability development, standards conformance, debugging, and technical exploration. The Developmental Piloting stage enables early user feedback on operational utility, integration and net-readiness, and security evaluations. The Operational Piloting stage facilitates any remaining security, interoperability, supportability, and usability assessments required to support an implementation decision.

When promoting a CM from one stage to the next, the JST, JCCD, and NECC SE assess the CM's maturity relative to its tailored TEC Criteria and decide whether it should be moved to the next stage, remain in its current stage for rework, or be removed. The FDCE Business Plan and Concept of Operations (CONOPS) lists criteria for CM promotion [Ref. 9, Tables 3 & 4]. In this regard, developers **SHOULD** actively participate in the following activities:

- Assertion against TEC Criteria in the FDCE
- Assessment of CM maturity against TEC Criteria for a particular stage
- Capability Provisioning Activities (CPAS) events to address shortfalls

2.4 Federated Development and Certification Environment and Related Registries

The FDCE is a virtual environment providing the policies, processes, and infrastructure that allow NECC CMs to be progressively refined, tested, and certified in increasingly rigorous situations leading to an operational environment.

The FDCE is also intended to facilitate the ongoing interaction and collaboration between NECC stakeholders throughout the entire development and certification process. Using the FDCE, developers can find NECC services, download relevant documentation, and locate points of contacts.

There are three FDCE sites: Piloting, Unclassified but Sensitive Internet Protocol Network (NIPRNet), and Secure Internet Protocol Network (SIPRNet). The FDCE Piloting site (www.fdce.net) gives an "early look" of FDCE and serves as a training ground for users. Developers **MUST NOT** enter any restricted government information into the Piloting site. The information there is not necessarily persisted or backed up.

The FDCE NIPRNet and SIPRNet sites (see Table 2) are the repositories for information related to NECC CMs. They will be used by various communities, including requirement engineers, program managers, developers, testers, certifiers, service providers, and operators to track status and maturity, to conduct testing and piloting, and to post documentation and evidence for the CMs. Information entered into the FDCE NIPRNet and SIPRNet sites will be persisted and backed up. Currently, developers need to register for three sites separately. Also, in order to improve the visibility of CM status, the TEC Criteria will be managed only on the FDCE NIPRNet site when possible.

CMs may exist in three stages in the FDCE. Those in the Development stage are under active development and consequently not stable. Those in the Developmental Piloting stage are

expected to be more mature, and hence more stable. CMs in the Operational Piloting stage are stable and being evaluated for operational deployment.

Developers MUST register all services under a CM with the FDCE before the CDR. In addition, developers MUST register eXtensible Markup Language (XML) schemas and service interface definition (Web Service Description Language, or WSDL file) in the DoD Metadata Registry (MDR) [Ref. 5].

To make the CM services visible, developers MUST also register service information with the Service Registry of NCES Service Discovery Service. The link to the WSDL file in the NCES Service Registry should point to the corresponding Uniform Resource Locator (URL) at the MDR. Figure 1 shows CM information and links in FDCE and these two related registries. Note that CM documents are currently hosted in the Defense Knowledge Online (DKO) FDCE Reference Library [Ref. 11]. For information on obtaining an account on DKO, see Section 3.2.3 in Ref. 10. Also, there are NIPRNet and SIPRNet versions for all these sites. Developers MUST ensure that CM information, including Internet Protocol (IP) addresses, is correct in these two network environment.

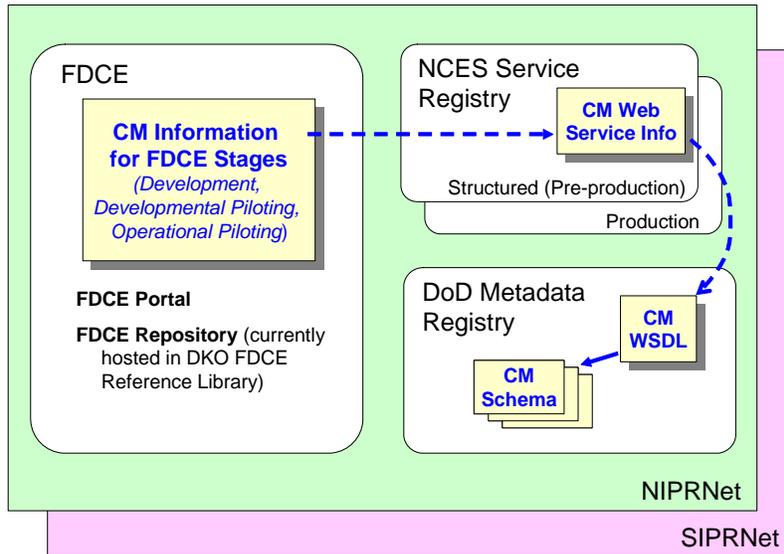


Figure 1: CM Information and Links in FDCE and Other Related Registries

The NCES Service Registry has production and pre-production sites. Table 2 gives the sites and links of FDCE and other related registries. Developers MUST ensure that a CM is registered with the correct registries corresponding to its maturity stage in Table 2. Note that the direct links to NCES Service Registry in Table 2 may be updated in the future. When that happens, the “link via DKO” should be used to navigate to the correct NCES Service Discovery Service. Additionally, developers need to follow the instructions in that DKO link to identify their user types (publisher, reader, etc.) for NCES Service Discovery Service.

Table 2: Sites and Links of FDCE and Other Related Registries

CM MATURITY STAGE	FDCE	MDR	NCES
NIPRNet			
Development, Developmental Piloting, Operational Piloting	FDCE NIPRNet [link]	MDR NIPRNet [link]	NIPRNet Structured (Pre-Production) [link] [link via DKO]
Operations	N/A	Same as Above	NIPRNet Production [link] [link via DKO]
SIPRNet			
Development, Developmental Piloting, Operational Piloting	FDCE SIPRNet [link]	MDR SIPRNet [link]	SIPRNet Pre-Production [link]
Operations	N/A	Same as Above	SIPRNet Production [link]

The FDCE User’s Guide [Ref. 10] describes how to register a CM and perform other functions at the FDCE. The FDCE Business Plan and CONOPS [Ref. 9] describes how the user community cooperatively executes their responsibilities in a federated, distributed, and shared environment to support rapid development, test, evaluation, and certification of CMs. The FDCE Reference Library [Ref. 11] also provides other information related to NECC.

Finally, the transition from pre-production to production (after approval by the Operational Test Agency) at the NCES Service Registry currently requires a manual step to re-register the CM service information (see Table 18 in Appendix I for contact information of NCES Service Registry support personnel). Detailed instruction on registering services with NCES is given in Ref. 6.

2.5 Configuration Management

The NECC Configuration Management Plan [Ref. 14] describes the roles and responsibilities, change control process, baseline management activities, etc. for configuration management. Items under configuration management include all artifacts delivered by developers in Table 13.

When developing a CM, developers MUST follow the versioning scheme below (which originates from the scheme used by Global Command and Control System Family of Systems (GCCS-FoS)).

A version number SHALL consist of a sequence of four integers, separated by decimal points (e.g., 1.2.3.4). The integers represent the following meanings:

- The first integer indicates a major release and a significant change in the architecture or operation of the CM. A major version would indicate that the specification may have changed in a non-backward compatible way.
- The second integer indicates a minor release in which new features have been added, but the fundamental architecture remains unchanged. A minor version would indicate that the specification has changed in a backward compatible way.

- The third integer indicates a maintenance release number. New features may be added but the emphasis is on optimization, feature enhancements, or modifications to improve stability and usability.
- The fourth integer indicates a bug related fix.

For configuration control of CMs, developers **MUST** use the following naming convention to identify their CMs:

AAA-CCC-MMM-v.v.v.v

where;

- AAA is the abbreviated mission area name for the CM. It contains one to three characters and is used in XML namespaces (see Section 3.2). The abbreviated names are: FPJ - Force Projection, FR - Force Readiness, I – Intelligence, SA - Situational Awareness, FE - Force Employment, FPT - Force Protection, and Cross Functions (CF).
- CCC is the acronym for the CM, such as JFS for Joint Force Synchronization and RAA for Readiness Assessment and Analysis. It has two or three characters and is also used in XML namespace (see Section 3.2).
- MMM is the acronym for the materiel solution under a CM. The number of characters is discretionary.
- v.v.v.v is the version number for the CM according to the NECC versioning scheme described above, i.e., 1.1.0.0 or 2.0.0.0.

The proper CM and service names will be provided in the CM Work Package. Developers **MUST** also register all services under a CM in the FDCE using the CM naming convention. Any change in those names **MUST** be made through a change request through the CCB.

For configuration management of multiple Capability Packages (CP) within a CM, two additional fields are used:

AAA-CCC-MMM-CP-O-v.v.v.v

Here the label “CP” (variable length) is an abbreviated Capability Package name designated by the developer/CPMO at their discretion. If a CM has only one Capability Package, this field shall be omitted. The character "O" indicates the guest operating system (see Table 6).

An example is: SA-ASM-C2CS-MSG-L-1.0.0.1, which stands for Situation Awareness (SA), Association Management (ASM), Command and Control Common Services (C2CS), Messaging (MSG), and Red Hat Enterprise Linux 4.0 (L).

Developers **MUST** provide detailed specifications in the Version Description Document (VDD) as to the COTS, GOTS, and guest operating systems for the Capability Packages within the CM. Some examples are given in Table 3. Note that the versioning scheme of COTS/GOTS software within a Capability Package is beyond the scope of this handbook.

Table 3: Capability Package Examples

SHORT NAMES	LONG NAMES AND PACKAGE REQUIREMENTS
FR-RAA-SOR-L-1.1.0.0	Force Readiness, Readiness Assessment and Analysis, SORTS 1.1.0.0 Enterprise Edition for Linux Capability Package Requirements: COTS: WebLogic Server 8.1SP5 Guest Operating System: Red Hat Enterprise Linux 4.0
FR-RAA-SOR-L-1.1.1.0	Force Readiness, Readiness Assessment and Analysis, SORTS 1.1.1.0 Enterprise Edition for Linux Capability Package Requirements: COTS: <u>WebLogic Server 9.2</u> Guest Operating System: Red Hat Enterprise Linux 4.0
FR-RAA-SOR-W-1.1.0.0	Force Readiness, Readiness Assessment and Analysis, SORTS 1.1.0.0 Local Edition for Windows Capability Package Requirements: COTS: WebLogic Server 8.1SP5 Guest Operating System: Windows Server 2003 SP2

The CM versioning scheme implies that different Capability Packages belonging to the same major and minor CM version SHALL interoperate seamlessly. When a CM with multiple Capability Packages undergoes maintenance or bug-fix version changes, it is possible that some of its Capability Packages need not be changed and therefore keep the same version number. This is allowed as long as all Capability Packages under the CM have the same major and minor version numbers. When a CM's major or minor version number changes, all Capability Packages within that CM must also reflect that version number change. Developers MUST clearly identify the Capability Packages in the VDD and describe the test results in the TPR for the CM consisting of new and old Capability Packages.

In the event that patches are required after delivery, developers MUST name them using the same versioning scheme, with patch number appended to the end of the version number. An example is: SA-ASM-C2CS-MSG-L-1.0.0.1P1.

Developers MUST adhere to the following file naming conventions for CM-specific documents:

<Title>_<Version>_<Date as yyyyymmdd>_<Status>.<extension>

Here "Title" may contain abbreviated mission area name, CM acronym, and program of record acronym (if applicable), separated by "_". For example, a final VDD for the Readiness Assessment and Analysis (RAA) CM under the Force Readiness (FR) mission area is,

FR_RAA_VDD_V1-0-0-0_20081005_FINAL.doc

Note that within the version number of a file name, hyphens (instead of periods) are used to separate the integers due to a restriction on file names uploaded to DKO.

2.6 Capability Package Delivery Process

For document delivery (including Capability Provisioning Events (CPE), User Free Play (UFP) and/or CM artifacts), developers MUST follow the process below:

1. CPMO/developer posts documents for JPMO review in the appropriate FDCE CM Library folder and sends an email notification to the Build Manager (see Table 18 in Appendix I for contact information).
2. JPMO reviews the documents and coordinates directly with the CPMO/developer to incorporate necessary changes prior to final acceptance of the documents.
3. Once a document is accepted as final by the JPMO, the CPMO/developer's Configuration Management personnel changes the filename to reflect a final status, posts the document to the appropriate folder, and sends an email notification to JPMO Configuration Management (see Table 18 in Appendix I for contact information).

Note that file naming conventions are given in Section 2.1.

For Capability Package delivering, developers MUST follow the process below:

1. Prior to Development Piloting, CPMO/developer's Configuration Management personnel checks in the "final" Capability Package artifacts (including documents and packaged code) into the Subversion repository of the NECC Configuration Management Site [Ref. 12] (or in the future the Definitive Service Library (DSL) at the Joint Technical Operations Control Capability (JTOCC)). In the repository, each CM has folders for relevant artifacts as dictated by the Work Package. Access to CM folders is restricted to corresponding CM developer's configuration management personnel.
2. Upon "check in" of final artifacts into the repository, an automatic email notification will be sent through DKO to JPMO Configuration Management (see Table 18 in Appendix I for contact information).

Note that for spiral delivery, tagging of a build will occur. Here a tag is a "snapshot" of a project in time with a name corresponding to the build.

All Capability Packages will be deployed to virtualized server environment based on VMWare ESX Server (see Section 3.1). Developers SHOULD follow the standard operating procedures for creating and deploying Capability Packages [Ref. 39]. The VMWare Converter 3.0 tool is used to package NECC software, COTS, and guest operating system to create VMWare image files (.vmx) for Capability Packages. As part of the Capability Package creation procedure, developers MUST lockdown the virtual machine [Ref. 39]. The naming of image files MUST follow the versioning scheme described in Section 2.5. Prior to delivery, developers SHOULD go through the deployment process for the Capability Packages and test the deployed Packages using a VMWare Server.

Developers MUST ensure that valid software licenses (not evaluation license) are used for COTS and guest operating system. Developers SHOULD consult the Software Licensing Management Plan [Ref. 13] when handling licensed software. This includes enterprise licenses, freeware, shareware, and open source.

2.7 Certification, Accreditation, and Information Assurance

CM Developers SHALL address all IA requirements allocated to their CMs in the Defense Information Assurance Certification and Accreditation Process (DIACAP) Implementation Plan (DIP). Developers MUST perform security engineering to include:

- Identification of data access control, protection, and transaction audit requirements with the data owners
- Implementation of controls necessary to satisfy identified requirements
- Certification and accreditation of their CM. Developers MUST address applicable architectural requirements documented in the NECC Increment 1 IA Architecture document [Ref. 2]

Developers MUST also follow DoD Security Technical Implementation Guides (STIGs) and Security Checklists [Ref. 15] to configure and lockdown software products used in CMs. Developers MUST also follow the guidance in the NECC Certification and Accreditation (C&A) Process document [Ref. 16] for certifying CMs. The document describes the C&A roles and responsibilities, accreditation boundary, identification of Information Assurance Officer, the Capability Security Plan, the Interim Approval to Test (IATT), and the Interim Approval to Operate (IATO) throughout the CM development stages. It also describes the use of the FDCE in the C&A process.

For each CM, the work package will designate the baseline IA requirements. These include Mission Assurance Category (MAC) and Confidentiality Level (CL) in accordance with DoD Instruction (DoDI) 8500.2 [Ref. 17] for NIPRNet and SIPRNet deployment and Protection Level (PL) in accordance with Director of Central Intelligence Directive (DCID) 6/3 criteria [Ref. 18] for Sensitive Compartmented Information (SCI) deployment.

The NECC Increment 1 IA Architecture document [Ref. 2] describes the key IA concepts, including the use of Public Key Infrastructure (PKI) for authentication, policy enforcement point, and policy decision point for CM services. Web Service developers SHOULD also consult the National Institute of Standards and Technology (NIST) Guide to Secure Web Services [Ref. 19]. For information on obtaining SIPRNet PKI certificates for developers, please refer to the PKI Reference Guides in Ref. 20.

A PKI server certificate is specific to a particular piece of hardware on the network. For NECC, such a server is a virtual machine (see Section 3.1 for details). Consequently, DoD PKI server certificates SHALL be installed on each virtual machine running CMs. The issuing authorities of such server certificates SHALL be based on network connectivity. For NIPRNet and SIPRNet operations, server certificates must be issued by a PKI Registration Authority (RA) or a Local Registration Authority (LRA). For testing on NIPRNet, JITC test certificates may be used for system development. Ref. 20 provides information related to PKI certificates.

When transferring CMs between locations (e.g., from a test site to a Defense Enterprise Computer Center (DECC)), a new server certificate is generally needed. However, under certain conditions, the same certificate may still apply. For example, if the domain and server names stay the same between locations and the server certificate does not contain IP address, the certificate will remain valid. Note, however, that JITC test certificates MUST not be used in production environments.

Based upon network connectivity, authentication preferences SHALL be in the following order (with the most desirable listed first). For operations on NIPRNet, use Common Access Card (CAC), DoD soft certificates, External Certificate Authority (ECA) certificates, or username/password. For SIPRNet, use DoD soft certificates or username/password. For testing on NIPRNet, JITC PKI test certificates may be used.

For authorization, it is expected that Role Based Access Control (RBAC) will initially be used. NECC will define a basic set of user roles, which can be extended by CMs. When the management of user attributes across NECC communities becomes mature, Attribute Based Access Control (ABAC) SHALL be implemented. ABAC enables authorized users (based on their attributes) not registered with a specific CM to invoke its services.

All REST (Representational State Transfer) type message exchanges between NECC consumers and service providers MUST use Transport Layer Security (TLS) protocol (which supersedes Secure Sockets Layer (SSL)) for authentication and/or encryption of data in transit. All NECC Simple Object Access Protocol (SOAP) transactions SHOULD use the NCES developed profile of WS-Security [Ref. 38].

For situations that are not covered in this handbook or the Increment 1 IA Architecture, developers SHOULD use good judgment and coordinate the proposed solutions with the NECC JPMO. The JPMO will update this handbook as appropriate to cover those situations and ensure interoperability between CMs in the future.

2.8 Net-Centric Data Strategy Compliance

As described in the NECC Increment 1 Data Architecture document [Ref. 2], NECC implements the DoD Net-Centric Data Strategy [Ref. 26] by focusing on the following tenets:

1. Make data visible. Developers MUST migrate from pockets of information to discovery patterns that enable data discovery across the enterprise.
2. Make data accessible. Developers MUST migrate from point-to-point proprietary access mechanisms to a common access mechanism that treats data assets as individually addressable data resources. In effect, the NECC data services establish a virtual shared space for data access.
3. Make data understandable. Developers MUST migrate from custom proprietary data of legacy systems to standardized/coordinated data interchange formats.

For NECC data services, Table 4 below gives the specific guidelines under these tenets. More details are in the NECC Increment 1 Data Architecture document [Ref. 2].

Table 4: Guidelines for Net-Centric Data Strategy Compliance

TENETS	GUIDELINES
Make data visible	<ul style="list-style-type: none"> • Store XML schemas and service interface definition (Web Service Description Language, or WSDL file) in the DoD Metadata Registry (MDR). • Register service information (including service endpoints) with the NCES Service Discovery. • Generate metadata in accordance with the DoD Discovery Metadata Specification (DDMS) [Ref. 27]. • Register metadata for data service with the NCES Content Discovery (Federated Search) and implement NCES Content Discovery/Search Web Service.

Make data accessible	<ul style="list-style-type: none"> • Provide CRUD (Create, Retrieve, Update, and Delete) and enumerated query operations in accordance with the web service access mechanisms in the NECC Increment 1 Software Architecture document. • Provide secure access in accordance with the NECC Increment 1 Security Architecture. • Provide a browser-based interface to interact with the data service. • Provide a browser-based interface for data subscription • (When applicable) Provide asynchronous access mechanism for data service.
Make data understandable	<ul style="list-style-type: none"> • Define data schemas using XML schema and namespace [see 0]. • Coordinate with the relevant Communities of Interest (COIs) in defining XML schemas. • Use the OpenGIS Geography Markup Language (GML) [Ref. 29] to tag geospatial/temporal data. This may involve using parts of GML relevant to a COI. • Tag data resources with Intelligence Community (IC) Information Security Marking (ISM) [Ref. 28] and follow a "tear-line" tagging model.

2.9 Operation and Support

During operation, services under a CM are monitored by the JTOCC for compliance with the Service Level Agreement (SLA). Relevant metrics are collected and alerts may be sent when a potential noncompliance occurs. Table 5 below gives the types of metrics defined in the SLA. Details are available from the NECC SLA Template [see the link in Table 13, Appendix C].

Table 5: Metrics Types for Services

METRICS TYPES	DESCRIPTIONS
Service Time	Response time for synchronous services. Delivery time for asynchronous services.
Scalability	Examples are user load and number of requests per second.
Availability	It includes planned maintenance and unplanned down time.
Reliability	It equals 1 - fault rate, where the faults are due to defects, rejected requests, message loss, etc.

The NECC Help Desk support concept is based on a multi-tier support structure [Ref. 2]. Tier 1 consists of round-the-clock support personnel. Tier 2 has subject matter experts for CMs. Tier 3 is CM developers who provide support during normal business hours and may be on call beyond those hours. If both Tiers 1 and 2 cannot resolve an issue or the resolution requires changes to a CM, the issue is escalated to Tier 3 and will be investigated based on its priority.

2.10 Training Materials

This section is applicable to developers who will produce training products in collaboration with Joint and Service training developers. Developers SHALL develop the products based on EMTs, Use Cases, common operational scenarios, and Operational Mission Threads (OMTs) that replicate realistic or practical conditions the individual user or unit will most likely encounter. The appropriate Joint or Service components will review, evaluate, and approve all training products before their distribution.

Under the guidance of the Single Service Training Manager (SSTM), materiel developers, along with supporting doctrine and training institutions, will create, redesign, or convert existing courseware to the Electronic Performance Support System (EPSS) format, Interactive Multimedia Instruction (IMI), Web-based training (WBT), or Computer-based training (CBT), as required, to comply with Advanced Distributed Learning-Registry (ADL-R) and Shareable Content Object Reference Model (SCORM) specifications.

The EPSS is an XML-based development and delivery environment for system documentation, online help, and embedded training. It is based on the GCCS-FoS Document Management Infrastructure (DMI) and is current being adapted to NECC requirements.

SCORM is an XML-based standard that allows training modules developed by different authoring tools to interoperate with a Learning Management System (LMS). See Appendix H for details.

If required by the work package, developers MUST follow the DMI Content Writer's Guide [Ref. 41] and use the EPSS to deliver training contents, including user's manuals (documentation and instructional slides), system administrator manuals, context sensitive help, and embedded just-in-time training. An example DMI user manual is given in Ref. 42.

Similarly, if required by the work package, developers MUST also develop a Functional Tutorial (FT), which are short simulations of a specific task or function. FTs may run in either observe mode or take action mode:

- In observe mode, the user views an animated simulation of a specific task or function. All the steps are viewed in a simulated environment and contain descriptive text and an audio overlay.
- In take action mode, the user performs the particular task or function in the simulated environment. Progression to the next step in the task is dependent on user input.

FTs are currently developed in Adobe Captivate and Flash. Examples can be found on the NECC Training Community of Interest (COI) portal hosted on the Joint Knowledge Development and Distribution Capability (JKDDC) [Ref. 43]. FT specifications and development guide will be provided in a separate document.

3 DESIGN AND DEVELOPMENT GUIDANCE

The following sections provide guidance for the design and development of CMs.

3.1 Server Environment

The server environment for NECC Increment 1 is based on a virtualized server platform (with the product VMWare ESX Server running on x86 servers). The platform, called a GCN, will be deployed at both enterprise sites (about four sites, supporting 500 or more users) and operational sites (about 800 sites, supporting 100 or fewer users). Operational sites often have disconnected, intermittent, or limited (DIL) network connection.

Each CM is packaged into one or more Capability Packages for deployment at the GCNs. A Capability Package contains software for a single CM only, including software developed specifically for NECC, COTS such as Web service containers and databases, and a guest

operating system. Developers **MUST** ensure that Capability Packages belonging to the same CM version are interoperable with each other.

Table 6 lists the approved operating systems for NECC. NECC will publish a list of pre-approved COTS infrastructure products. Prior to the publication of the list, developers may select any COTS infrastructure products that work with the operating systems in Table 6.

Table 6: Approved Operating Systems for NECC

SHORT NAMES	OPERATING SYSTEMS
L	Red Hat Enterprise Linux v.4.0
S	Solaris x86 v.10
W	Windows Server 2003

Each virtual machine is dedicated to a particular Capability Package. The configuration parameters are given by Table 7. Note that NECC may specify upper bounds for these parameters for certain types of GCNs. Developers **MUST** specify the virtual machine configuration for each Capability Package (e.g., as part of the information table for provisioning CMs [see Table 14] in the Administration Manual).

Table 7: Virtual Machine Configuration Parameters

PARAMETERS	EXAMPLES
Processor	2 CPU (Central Processing Unit)
Memory	2 GB (Gigabyte)
Hard disk space	20 GB (Basis of estimate: 820 MB for WebLogic with Java Runtime Environment, 1.5 GB for Windows Server 2003, 10 GB for temporary files)

More details about the GCN specification are in the NECC Increment 1 Physical Architecture document [Ref. 2]. Information about virtual machine is in Ref. 40.

3.2 Schema Development

In developing NECC XML schemas, developers **MUST** follow the layered model described in the NECC Increment 1 Data Architecture document [Ref. 2]. The core layer contains common elements for all CMs. Those elements are from the DDMS [Ref. 27], the IC ISM [Ref. 28], and the Open GML [Ref. 29].

Developers **SHOULD** import these common schemas and use or extend them to define CM specific elements. For instance, the Information Security Markings are used by applying the IC ISM schema as an attribute group to selected XML elements. An example is given in Table 16 of Appendix F. Below are instructions and recommended best practices on schema development.

Separation of schema definitions – developers **MUST** define schemas for a service in separate schema files from the service interface definition (e.g., WSDL file). This allows the schemas to be reused and potentially merge with other similar schemas as appropriate. Developers **MUST** ensure that the tools they use for schema development support this requirement.

Namespace convention – XML namespaces in the DoD Metadata Registry have the format:

```
http://metadata.dod.mil/mdr/ns/<GovernanceNamespacePrefix>/<AAA-CCC>/<version>/[name]
```

Here, a Governance Namespace Prefix is controlled by the relevant governing agent for a Community of Interest (COI). For NECC, the following namespace SHOULD be used:

```
http://metadata.dod.mil/mdr/ns/NECC-C2/<AAA-CCC>/<version>/[name]
```

However, in the cases where namespaces are already defined by COIs, developers MUST coordinate with the COIs to determine which Governance Namespace should be used. Developers SHOULD also involve the relevant COIs in defining XML Schemas and vocabularies used for data exchange.

The other required fields in the namespace format are AAA-CCC and version. Here as in Section 2.5, AAA is the abbreviated mission area name (e.g., FR for Force Readiness) and CCC is the acronym for the CM (e.g., RAA). The version number is the same as that in Section 2.5 (e.g., 1.1.0.0). Note that namespace is defined at the CM level for data exchanges between the materiel solutions in a CM, or across CMs.

The optional field "name" indicates either a type name in a schema (as in Mission Essential Task List (METL)) or a service name (as in Readiness Information).

Schema version - the root element of the schema definition SHOULD have a "version" attribute with a value matching the version number in the namespace. Table 16 of Appendix F gives an example.

Naming Convention - Naming conventions will be controlled by the COIs that developed and/or control the vocabulary or XML schemas being leveraged. Developers should be prepared to deal with multiple naming conventions, particularly when XML schemas from multiple COIs are involved. Developers MUST ensure that codes for CMs do not assume or rely upon a particular naming convention.

Universal Unique ID (UUID) – developers MUST use the string representation of a UUID defined in Ref. 30. An example UUID string is

```
urn:uuid:da726315-bc91-430b-9ed8-ce5ffb858a92
```

The lowercase hexadecimal numbers are used following "urn:uuid:". Furthermore, for security reasons, if version 1 UUID is used, developers MUST use a randomly generated value for the node ID instead of the machine address (see Section 4.5 of Ref. 30).

3.3 Data Service Development

The NECC Increment 1 Software Architecture document [Ref. 2] provides a list of technical standards that govern the implementation of NECC CMs. For synchronous (request-response type) data services, developers MUST follow the enterprise resource pattern (described in the NECC Increment 1 Software Architecture document [Ref. 2]) and use the Context Data Source Adapter (CDSA) Specification [Ref. 31], which is a profile of the WS-Management standard. Additionally, to indicate a service is operational, developers MUST implement the heartbeat function as outlined in section 7.2.5 of the WS-Management specification for all delivery modes.

For geospatial data, developers **MUST** use the Open Geospatial Consortium (OGC) standards such as WMS (Web Map Service), SLD (Styled Layer Descriptor), WFS (Web Feature Service), and WCS (Web Coverage Service) [Ref. 32].

Really Simple Syndication (RSS) is popular for lightweight event notification. It is used similarly in NECC. However, developers **MUST** not use RSS as a replacement for the more robust messaging mechanism provided by the NECC Enterprise Service Bus. A RSS Best Practices Profile is being developed by the RSS Advisory Board [Ref. 33]. It also provides a feed validator. Note that the OGC standards and RSS are REST-style service standards.

For date-time values in the input/output of CM services, Developers **MUST** use the Coordinated Universal Time (UTC) as specified in the NECC Increment 1 Data Architecture document [Ref. 2]. Table 15 in Appendix E summarizes the usage of these and other standards.

3.4 Human Computer Interface and Asynchronous JavaScript and XML (AJAX)

Unless otherwise specified in the work package, developers **MUST** provide presentation services for all CMs. Developers **SHOULD** follow the guidelines in the Web Portal Design Guide [Ref. 21] in designing Web-based user interfaces. This ensures a consistent user experience across different NECC presentation services and enhances their usability. The main topics of the guide are Web application concepts and design consideration for portals, common information views, interface controls, and user interaction. It also recommends that a human-factor engineer be involved early in the user interface design.

The Web application concepts section recommends a shallow structure for applications. The user should reach important information in a maximum of three jumps. There should be a logical mapping between information structure and available navigation controls. The page layout should contain a banner, navigation area, content area, and footer. The Guide gives additional guidance on application structure, portal layouts, and how to reduce memory load on users.

Another important section in the guide are common information views and interface controls. These cover charts, forms, information only, maps, search, and table views. The guide also recommends keeping all stylistic elements in a Cascade Styling Sheet (CSS) separate from the page content. The guide also discusses user interaction in Web pages. This includes pointing device, keyboard, navigation, selection and activation, feedback, and confirmation. Developers **MUST** also use the Human Computer Interface (HCI) Checklist in the guide to review their design.

As to technical standards for Web-based HCI, developers **MUST** use the list of standards in the NECC Increment 1 Software Architecture document [Ref. 2]. Validation tools for these standards are in Ref. 22.

A recent technique that enhances user interface interaction is AJAX. It uses a combination of existing technologies to deliver rich user experience similar to desktop applications through a Web browser. With AJAX, part of the Web page is updated or constructed dynamically when a user event occurs. On the other hand, the data communication between the browser and the server may happen asynchronously relative to the event. Views such as text field, table, chart, and map may be enhanced with AJAX.

For NECC, presentation services such as those in Situational Awareness may apply AJAX to enhance end user experience and productivity. Appendix G provides some examples for selected views.

AJAX coding requires proficiency in JavaScript, CSS, and Document Object Model (DOM), which may be different in different browsers. The preferred development strategy is to use an existing AJAX tool kit or an AJAX framework with ready-made user interface components [Ref. 23].

In addition to the general HCI guidance above, some best practices when designing AJAX Web applications are as follows:

- **JavaScript and accessibility** - AJAX requires JavaScript to be enabled. Developers SHOULD include a note in the initial Web page to inform users about this requirement. Alternatively, a developer could design the page such that users with JavaScript disabled can still view a working version (e.g., by using the Hyper Text Markup Language (HTML) "NOSCRIPT" element, or following a progressive enhancement approach [Ref. 24]). Test the page with a text only browser such as Lynx. This is important for sites that are required to be Section 508 compliant and to provide access for disabled people.
- **Back button** - a Web page with AJAX may behave like a full application. Thus if a user clicks the back button, the page will be unloaded and all information about the state of that application will be lost. A warning SHOULD be given when the user tries to leave such a page with state information about the application. Alternatively, developers SHOULD use a dedicated window without browser controls to contain the application, or split the major sections of the application into several pages with clear indication when a user event triggers an unload of the application.
- **Bookmark** - similarly to the back button issue, bookmarks for a Web page with AJAX may not direct browsers back to the desired state of the AJAX application. If there are logical links to certain states of the AJAX application, they SHOULD be designed into the application, e.g., by dynamically constructing a full Uniform Resource Locator (URL) to them.
- **Chatty communication** - AJAX allows user events to trigger requests to the server. Care must be taken to avoid excessive communication. Developers may consider batching multiple requests into a single request.
- **Staged transfer** - for a rich Web application may take a long time to transfer the JavaScript codes. Developers SHOULD stage the application by sending the minimal code needed for rendering the page first. Then send the rest in a priority order (e.g. by using the "SCRIPT" element in the HTML document object model to dynamically control the loading of JavaScript through a timer). This helps improve the perceived performance of the application.
- **Security** - developers MUST follow the same rules and due diligence as standard Web applications in developing AJAX applications. Some common vulnerabilities requiring attention are: cross site scripting, privilege escalation via DOM property overrides, and session hijack.

Taking AJAX applications one step further, one may connect to multiple Web services to obtain data and to remix them to build new applications. This is part of the vision for Web 2.0 [Ref.

25], which is a term often applied to a perceived ongoing transition of the World Wide Web from a collection of websites to a full-fledged computing platform serving web applications to end users. This is similar to DoD's vision of providing net-centric capabilities through GIG Enterprise Services (GES). The CMs in NECC fully enables such a vision, and the vast number of emerging Web 2.0 applications can be good references.

3.5 Net-Centric Enterprise Services

NCES is a program that provides Core Enterprise Services (CES) [Ref. 35]. Developers SHOULD use these services, which are currently offered through the Service Oriented Architecture Foundation (SOAF), Content Discovery and Delivery, Enterprise Collaboration, and Enterprise Portal.

An Early Capability Baseline (ECB) of NCES is already available [Refs. 35 and 36]. The MOA between NECC and NCES (Ref. 37) lists the following release schedule for NCES: v0.1.1 in September 2007, v0.2.0 in February 2008, and v1.0.0 in January 2009. Also, NCES is expected to achieve Initial Operational Test and Evaluation (IOT&E) by the third quarter of FY08.

NCES will have two primary sites on the NIPRNet and two primary sites on the SIPRNet, located at the DECCs in Columbus and San Antonio. They are operational 24 x 7 x 365. Processing load is balanced between the primary sites, which also have data replication/mirroring between them. NCES will also have integration labs within the DECC environments (SIPRNet and NIPRNet).

Table 8 lists the Core Enterprise Services and their usages by CMs. It also shows the availability of the services by NCES versions. More technical information about NCES, including architecture and Service Development Kits (SDKs) is in Ref. 35.

Table 8: Core Enterprise Services provided by NCES and their usages by NECC CMs

CORE ENTERPRISE SERVICES	DESCRIPTION	CM USAGE GUIDANCE	AVAILABILITY IN NCES VERSIONS
Service Security	Provides the following services: <ul style="list-style-type: none"> • Robust Certificate Validation Service • Attribute Service 	Developers MUST use the Robust Certificate Validation Service to assure that DoD PKI Certificates have not been revoked.	v0.1.1, v0.2.0, v1.0.0

CORE ENTERPRISE SERVICES	DESCRIPTION	CM USAGE GUIDANCE	AVAILABILITY IN NCES VERSIONS
Machine-to-Machine (M2M) Messaging	<ul style="list-style-type: none"> Creates/updates/deletes/lookup/search topics Publishes/subscribes messages Supports synchronous and asynchronous messages with point-to-point and one-to-many patterns Supports reliable messaging Provides cross (SOA) boundary Messaging Has SOAP-based Web service interface and portlet Graphical User Interface (GUI) 	<p>NECC will establish an Enterprise Service Bus (ESB) that is compatible with and leverages to the maximum extent possible NCES M2M Messaging (e.g. using it as a message broker for the NECC ESB). The NECC ESB will provide adapters (e.g., JMS adapter) for integration with such message brokers.</p> <p>Asynchronous message exchange across CMs MUST use this ESB when it is developed. CMs may use other ESB products for internal exchanges. CMs may also use NCES messaging for publishing/subscribing messages beyond NECC.</p>	v0.1.1, v0.2.0, v1.0.0
Service Discovery	<ul style="list-style-type: none"> Registers/Publishes/Search Service [based on UDDI] Has SOAP-based Web service interface and portlet GUI 	<p>Developers MUST register all operational CM services with Service Discovery. NECC consumers SHALL use it to find CM services.</p>	v0.1.1, v0.2.0, v1.0.0
Mediation	<ul style="list-style-type: none"> Transformation - Extensible Stylesheet Language Transformation (XSLT) based transformations of data Adaptation - translate information protocols from popular standards (i.e., standards and proprietary based) to other popular information protocol Orchestration - facilitates interaction between services by allowing integration developers to register, via a web-based user interface, a standards-based instruction set to define a workflow. Has SOAP-based Web service interface and portlet GUI 	<p>CMs may use these services as needed.</p>	Only XSLT based transformation of data currently available for ECB. Others will be available in v1.0.0.
Metadata Registry	<ul style="list-style-type: none"> Finds Extensible Stylesheet Language Transformation (XSLT) stylesheets from the DoD Metadata Registry Searches the Registry via SOAP-based Web service 	<p>Developers MUST register all metadata (e.g. XML schemas) developed under a CM with the DoD Metadata Registry. This service enables search on the Registry for NECC and other metadata.</p>	v0.1.1, v0.2.0, v1.0.0

CORE ENTERPRISE SERVICES	DESCRIPTION	CM USAGE GUIDANCE	AVAILABILITY IN NCES VERSIONS
Enterprise Catalog	<ul style="list-style-type: none"> Allows submission of catalogs (which are DDMS compliant). Indexes those catalogs and works with Federated Search to make data searchable by the entire enterprise. 	Data service CMs have the option to submit catalogs, which are indexed and made searchable through Federated Search.	v0.1.1, v0.2.0, v1.0.0
Federated Search	<ul style="list-style-type: none"> Searches data on multiple, distributed sources (databases, catalogs or search engines) Aggregates the results before returning them to the requestor 	NECC consumers SHOULD use Federated Search to find links to data from data service CMs. Retrieval of the actual data is still performed by those CMs. CMs maintain control of their data resources.	v0.1.1, v0.2.0, v1.0.0
Enterprise Service Management (ESM)	<ul style="list-style-type: none"> Collects service metric (including throughput, fault rate, response time, availability, status, and usage) Publishes the metrics into a Universal Description, Discovery, and Integration (UDDI) Service Discovery Catalog Sets thresholds and publishes threshold failures Has SOAP-based Web service interface and portlet GUI 	Instead of using NCES ESM, CMs hosted at a GIG Computing Node will be managed according to the Joint Technical Operations Control Capability (JTOCC) Execution Plan.	N/A

As described in the NECC Increment 1 IA Architecture document [Ref. 2], incoming messages are intercepted by a handler installed with the CM service provider. The handler acts as a Policy Enforcement Point to identify and authenticate the requestor.

The NCES Robust Certificate Validation Service is invoked during this process to authenticate the requesting user or server. Subsequently, a Policy Decision Service authorizes the request based on predefined access control policy, the identity of the requestor and, possibly, the attributes of the requestors. If attributes are needed to make an authorization decisions, the NCES Attribute Service may be invoked along with the Local Attribute Service.

The future NECC User Management CM will offer a Policy Enforcement Point, a Policy Decision Service, a Local Attribute Service, and a Policy Administration Service. The implementation will be based on the NCES Service Security (SS) specifications [Ref. 38].

For services that implement the SOAP standard, developers SHOULD use the NCES Service Security profile of WS-Security. Developers SHOULD use the NECC User Management CM when it is available, or they will need to implement similar functionalities provided by that CM. The User Management CM will not be available for NECC Milestone C. CMs built for Milestone C SHALL implement their own authentication and authorization mechanisms consistent with the NCES Service Security Interface Specification and Design Specification [Ref. 38].

For services that implement the REST style (e.g., geospatial data and RSS services), developers need to build their own Policy Enforcement Point in order to use NCES Service Security. The

Policy Enforcement Point may invoke the User Management CM when it is available to obtain attributes, relevant policies, and authorization decisions using its published interfaces. Until the User Management CM is available, developers SHALL implement their own authentication and authorization mechanisms for CMs using the REST style. For performance reasons, the Policy Decision Service may be collocated with the CM service provider

3.6 Migrating to the NECC SOA

Many of the NECC CMs are migrated from GCCS-FoS. This involves, for example, migration from an N-tier architecture to the NECC SOA. The CMs identified for Increment 1 already separate the tiers into services, including presentation, visualization, and data. Since data exchanges across these CMs go through networks, care must be taken to avoid chatty communication. Common functions, such as redirection, are handled by dedicated CMs, rather than in individual applications. Additionally, CMs need to be instrumented to allow service management. Guidance for these topics is briefly discussed below.

Chatty and Chunky Communication – an existing data tier generally offers fine-grained data access through database queries. If this is wrapped into a straightforward Web service, it will likely take a consumer many calls to retrieve the necessary information. This results in chatty communication that leads to poor end-user experience when network latency is significant. On the other hand, a large (chunky) data set returned by the service increases the demand on the consumer's available bandwidth and processing power. Developers SHOULD analyze the key use cases for a CM in collaboration with expected service consumers to identify the optimal granularity for data exchange and design the service interface and data schema accordingly. Since a CM is no longer confined to a vertically integrated N-tier architecture, it may be necessary to implement different granularity options for different consumer groups.

Redirection CM – this CM allows the consumers of an NECC service to use a single global URL to invoke the service. The consumers are automatically routed to the appropriate GCN hosting the service (see the NECC Increment 1 Software Architecture document [Ref. 2]). For Increment 1, the Redirection CM will only apply to services hosted by Enterprise GCNs. Developers MUST ensure that CM data are adequately replicated and synchronized across the Enterprise GCNs, so that consumers receive the correct data regardless of the Enterprise GCNs fulfilling their requests.

CM Instrumentation – developers SHALL provide management interfaces to their CMs in order to give the NECC JTOCC the ability to remotely monitor, administer, and maintain them. Additionally, to indicate a service is operational, developers MUST implement the heartbeat function as outlined in section 7.2.5 of the WS-Management specification for all delivery modes.

ESB Integration - in most cases, the best practice is to have only one ESB in an enterprise. However, there are several tactical reasons for having more than one ESB, including for example, multiple governance bodies, different funding models, alignment by organizational units, distributed geographic locations, different business strategies, and multiple ESB technologies. Multiple ESB implementations may be temporary, or it may be a permanent choice if the enterprise is bound with the above issues. NECC will employ a brokered/federated ESB topology. Each ESB will integrate with the broker ESB (the NECC ESB) rather than with individual ESBs. Implementation details, such as routing rules, are encapsulated by the broker ESB. This creates a loose coupling so changes in one ESB will be less likely to affect the others.

APPENDIX A – REFERENCES

The NECC home site [link], currently hosted on the DKO, provides references and ongoing information about the program (documents, meeting announcements, contact information, etc.). The table below lists by category the reference documents cited in this handbook and their links.

Table 9: Reference Documents for the Handbook

DESCRIPTIONS AND LINKS	FILENAMES/ NOTES
Program level and architecture	
1. NECC Capability Development Document (CDD), Increment 1 [link]	N/A
2. NECC Increment 1 Architecture Documents (Overview, Physical, Software, Data, Information Assurance, Technical Operation, and Architecture-Driven Requirements) [link]	N/A
Systems engineering	
3. NECC Systems Engineering Plan [link]	N/A
4. NECC Request for Deviation Procedure [link] (under the folder Configuration Management Library / NECC Specific Artifacts)	N/A
5. DoD Metadata Registry [link]	XML_Registry_User_Guide.doc
6. NCES Service Discovery & Publishing Process, v.0.7, 21 November 2007	NCES Service Registration Process v0-7.doc
7. NECC Test, Evaluation and Certification Criteria [link]	N/A
8. NECC Test and Evaluation Master Plan [link]	N/A
9. FDCE Business Plan and Concept of Operations [link] (under the folder FDCE User & Reference Docs / CPAS CONOPS)	N/A
10. FDCE Users Guide [link] (under the folder FDCE User & Reference Docs / Users Guide)	N/A
11. FDCE Reference Library [link]	N/A
12. NECC Configuration Management Site (Subversion repository) [link]	N/A
13. NECC Software Licensing Management Plan (being developed)	N/A
14. NECC System Development and Demonstration Configuration Management Plan [link]	N/A
15. DoD Security Technical Implementation Guides [link] and Checklists [link] .	N/A
16. NECC Certification and Accreditation Process [link] .	N/A
17. Information Assurance (IA) Implementation, DoDI 8500.2, 6 February 2003 [link]	N/A
18. Protecting Sensitive Compartmented Information within Information Systems, Director of Central Intelligence Directive 6/3, 23 December 2003 (Unclassified//For Official Use Only)	N/A

DESCRIPTIONS AND LINKS	FILENAMES / NOTES
19. Guide to Secure Web Services, NIST Special Publication 800-95, August 2007 [link]	N/A
20. PKI Certificate References [link] (under the folder FDCE User & Reference Docs / PKI and SIPR Cert Reference Docs)	N/A
Design and development	
21. Web Portal Design Guide, v1.1 Final, July 2006 [link] .	Web Portal Spec v11 Final.doc Web Portal Checklist v11 Final.xls
22. W3C Validation Service - for Extensible Hypertext Markup Language (XHTML) and Cascading Style Sheet (CSS) [link] .	N/A
23. AJAX Design Strategies, October 2006 [link] . Some popular AJAX tool kits are Prototype, Dojo, Rico, and script.aculo.us. Some AJAX framework tools are: Backbase, Google Web Toolkit, Yahoo! UI Library, DWR, JackBe, Microsoft Atlas, General Interface, and OpenLaszlo. Some coding examples [link] .	N/A
24. Hijax: Progressive Enhancement with Ajax [link] .	N/A
25. What is Web 2.0 [link] .	N/A
26. Data Sharing in a Net-Centric Department of Defense, DoD Directive 8320.02, December 2, 2004 [link] .	N/A
27. DoD Discovery Metadata Specification (DDMS) specifies a set of information fields for the discovery of data or service assets across DoD [link] .	<i>Note: For existing schemas defined with DDMS v.1.3 or v.1.4, developers MUST migrate them to the current v.1.4.1.</i>
28. Intelligence Community (IC) Information Security Marking (ISM) is a set of XML attributes for tagging data elements [link] . The schema is also in the DoD Metadata Registry under the Intelligence namespace.	IC-ISM-v2.xsd (inside DDMS_v1.4.1.zip)
29. Geography Markup Language (GML) v. 3.2.1 [link] .	N/A
30. A Universally Unique IDentifier (UUID) URN Namespace, July 2005 [link] .	N/A
31. Context Data Source Adapter (CDSA) Specification, v.2.0, 2 November 2007 [link] .	N/A
32. Web Map Service (WMS) [link] .	N/A
33. Really Simple Syndication (RSS) Best Practices Profile [link] and validator .	N/A
34. National Security Agency (NSA), XML Schema Guidance for Cross Domain Security Policy Enforcement, version 0.9.0, 24 July 2006	N/A
35. NCES Developer Community [link] .	N/A
36. NCES Product Line Workspace [link] . (Note that the Metadata Services information there is for version 5, rather than the current version 6.)	N/A
37. NECC-NCES Memorandum of Agreement (MOA), DISA Agreement Number ITL-07-011, September 2007.	N/A
38. NCES Service Security Interface Specification and NCES Service Security Design Specification [link] .	N/A

DESCRIPTIONS AND LINKS	FILENAMES / NOTES
Virtual machine	
39. NECC Capability Package Virtual Machine Creation and Deployment Standard Operating Procedures (SOPs), April 2008 (draft).	N/A
40. Virtual Computing Security Technical Implementation Guide, Version 1, Release 0.1, 13 April 2007 [link] (under the folder AWG Products / Developers Handbook / References)	N/A
Training content development	
41. Unified Systems Manual - Documentation Management Infrastructure (DMI), Content Writer's Guide, Release 1.0, December 2003.	USM-DMI Content Writer's Guide-v.1.doc
42. Example DMI user manual.	DMI User Manual Example.xml
43. NECC Training Community of Interest website [link].	N/A
44. SCORM® 2004 is available at the Advanced Distributed Learning Web site [link].	N/A
45. Comparison of SCORM-Compliant Training Development Software, March 29, 2007	LMS_SCORM_Tools_Comparison-2007.pdf

APPENDIX B – CROSSWALK WITH INCREMENT 1 ARCHITECTURE

Table 10 lists the topics in Increment 1 Architecture documents and their corresponding sections in this handbook. (Note: when using the hyperlinks below, if necessary, activate web-like back arrow feature by right clicking your mouse in the light blue area beside the formatting toolbar above>scroll down the popup box and click on “Web” (an orange box w/ a checkmark will appear beside the word “Web”. A toolbar will then appear with green directional arrows that you can use to get back to this table)

Table 10: Crosswalk with Increment 1 Architecture Documents

TOPICS IN INCREMENT 1 ARCHITECTURE DOCUMENTS	HANDBOOK SECTIONS
Physical Architecture	
GIG Computing Node	3.1
Virtual machine and configuration parameters	3.1
Software Architecture	
Capability Package	3.1
Presentation service	3.4
Synchronous data service using enterprise resource pattern and WS-Management	3.3
Synchronous data service for geospatial data using OGC standards	3.3
Synchronous data service exposed using RSS	3.3
Asynchronous data service using Enterprise Service Bus	3.3 and 3.6
Registration with NCES Discovery Service	3.5
Compliance with Redirection CM	3.6
Caching and proxy mechanism for Disconnected, Intermittent, and Limited Communications operation	N/A
Data Architecture	
Three-layer data model (core, Community of Interest , CM-specific)	3.2
Granularity of resources	3.2 and 3.6
Metacard content and generation	3.2
DoD Metadata Registry	3.5
Compliance with NCES Discovery Service	3.5
Tagging data with IC ISM	3.2
Tagging geospatial data with GML	3.2
Information Assurance Architecture	
Policy decision point for CM services	2.7
Role and attribute based access control	2.7
Confidentiality for data in transit	2.7
Cross domain XML data	2.7
Technical Operations Architecture	
Tier structure and process flow	2.9
Service runtime governance and monitoring	2.9

APPENDIX C – SYSTEM ENGINEERING REVIEWS AND DELIVERABLES

Figure 2 shows the nominal CM development milestones and deliverables. Details about the SE reviews are given in Table 11. Deliverables from developers are in Table 13.

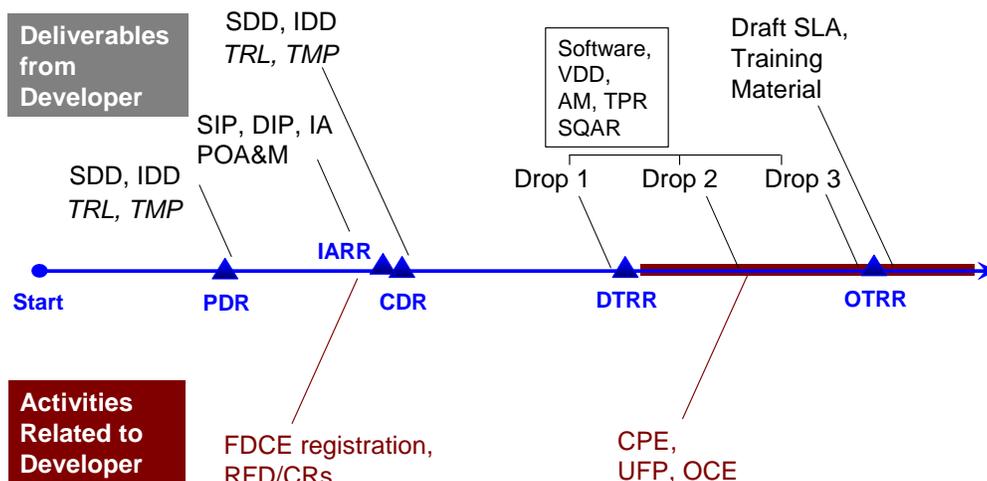


Figure 2: Nominal CM Development Milestones and Deliverables

Table 11 lists the required artifacts for SE reviews of a CM. The artifacts marked with “D” in Table 11 are the responsibility of developers. Developers MUST deliver them to the SE Review Manager (see Table 18 for contact information) before a formal review can be scheduled. Also, note that the Operational Test Readiness Review (OTRR) will only be conducted for level 3 and 4 Operational Test events.

Table 11: Artifacts for Systems Engineering Reviews

REQUIRED ARTIFACTS	
Preliminary Design Review (PDR) [SEP Step 7a]	
Completed NECC Design Review Checklist (See Table 12, PDR portion)	D
CPMO Validation	
JPMO Approval	
PDR Formal Meeting Minutes	
Information Assurance Readiness Review (IARR) [SEP Step 7b]	
IA Registration in FDCE	D
Certification & Accreditation Documentation	D
DIACAP Implementation Plan	D
IA Plan of Actions and Milestones (POA&M)	D
DAA IATO/ATO Approval	D
Critical Design Review (CDR) [SEP Step 7c]	
Completed NECC Design Review Checklist (see Table 12, CDR portion)	D
CM Registration with FDCE	D
CPMO Validation	
JPMO Approval	
CDR Formal Meeting Minutes	

Developmental Test Readiness Review (DTRR) [SEP Step 8]
CPMO Checklist
JPMO T&E Checklist
JPMO Logistics Checklist
JPMO SE Checklist
DTRR Certification Sheet
Operational Test Readiness Review (OTRR) [SEP Step 8]
CPMO Checklist
JPMO T&E Checklist
JPMO Logistics Checklist
JPMO SE Checklist
OTRR Certification Sheet

Table 12 lists the review items for PDR and CDR. Developers SHOULD consult this checklist when preparing for the reviews.

Table 12: NECC Design Review Checklist

REVIEW ITEMS	PDR	CDR
Has the developer demonstrated an understanding of how the CM design fits into the overall NECC SOA?	<input type="checkbox"/>	
Does the design adequately represent the high-level functions of the CM described in the CM Specification?	<input type="checkbox"/>	
Has the developer documented the logical (functional) and physical architecture at a high level (cf. the Software Design Description template)?	<input type="checkbox"/>	
Has the developer documented the logical (functional) and physical architecture at a detailed level?		<input type="checkbox"/>
Has the developer described the interfaces for all services under a CM? This includes inputs, outputs, and operations.	<input type="checkbox"/>	
Has the developer defined all operations and associated schemas under those services (e.g. using the Web Services Description Language)? (See the Interface Design Description template for details.)		<input type="checkbox"/>
Are the data exchanges and services compliant with the standards mandated by NECC?		<input type="checkbox"/>
Has the developer described all external interfaces related to the CM?	<input type="checkbox"/>	
Has the developer defined all external interfaces and their data exchanges?		<input type="checkbox"/>
Has the developer described the presentation services in the context of the use cases for the CM?	<input type="checkbox"/>	
Has the developer described user interface design with mock-up screens of the user interfaces? Has a human-factor engineer been involved in the user interface design?		<input type="checkbox"/>
Has the performance of the CM been estimated and does it meet the requirements?		<input type="checkbox"/>
Has the CM been registered with the FDCE? Have the relevant metadata and schemas been registered with the DoD Metadata Registry? Have the services been registered with the NCES Service Registry?		<input type="checkbox"/>
Has the developer resolved the issues identified during the PDR? Have the RFDs for the CM been approved by the CCB?		<input type="checkbox"/>

REVIEW ITEMS	PDR	CDR
Has the developer identified all relevant components of existing systems (e.g. GCCS-FoS) that will be migrated to the CM (if applicable)?	<input type="checkbox"/>	
Has the developer identified COTS/GOTS software to be used with the CM?		<input type="checkbox"/>
Has the developer identified the guest operating system and virtual machine configuration appropriate for the CM?		<input type="checkbox"/>
Has the developer summarized the CM's Technology Readiness Level evaluation and described the plan for technology maturity?	<input type="checkbox"/>	<input type="checkbox"/>
Has the developer described the hosting strategy at GIG Computing Nodes?		<input type="checkbox"/>

Table 13 lists the deliverables from developers. The Developmental Work Package for a CM specifies the deliverables and due dates specific to the CM. When applicable, the standard DoD Document IDs are provided as specifications for the deliverables. For some deliverables, NECC specific templates have been developed. In this case, developers SHOULD use these templates by downloading them from the link provided in the table.

Table 13: Deliverables from Developers

#	DELIVERABLES	DESCRIPTIONS
1	<p>Software Design Description (SDD) [Template] (under the folder AWG Products / Developers Handbook / CURRENT)] [DI-IPSC-81435A]</p> <p>Delivery: (1) draft due 5 days after PDR; (2) final due 5 days after CDR</p>	<ul style="list-style-type: none"> Documents the logical (functional) and physical architecture, as well as hosting strategy at the planned Global Information Grid (GIG) Computing Nodes. SHOULD use standards such as Unified Modeling Language (UML) and relevant DoD Architecture Framework (DoDAF) products (such as SV-1 and SV-2 in UML representation). MUST include all external interfaces not covered by the IDD. The SDD SHALL be reviewed in the preliminary and critical design reviews.
2	<p>Interface Design Description (IDD) [Template] (under the folder AWG Products / Developers Handbook / CURRENT)] [DI-IPSC-81436A]</p> <p>Delivery: same as that for Software Design Description</p>	<ul style="list-style-type: none"> Describes the interfaces for all services under a CM for service consumers/end-users. For Web services, MUST include documentation for all operations defined under those services (e.g. in the Web Services Description Language files). MUST provide description of the sample consumer codes in the Software End Items. For presentation services, MUST include designs for all user interfaces. Developers SHOULD follow the guidelines in Section 3.4. The IDD SHALL be reviewed in the preliminary and critical design reviews.

#	DELIVERABLES	DESCRIPTIONS
3	<p>Software End Items</p> <p>Delivery: (1) 5 days prior to DTRR (2) 30 days after DTRR (or N days relative to CPE/UFPO/OCE) (3) 5 days prior to OTRR</p>	<ul style="list-style-type: none"> • Compiled code packages (e.g. a Web Application archive, or WAR file) for each Capability Package under the CM. Each Capability Package is to be delivered with a guest operating system and an optional set of supporting Commercial-Off-the-Shelf (COTS) infrastructure software. Include any required software libraries. • MUST include a completed GCN pre-deployment checklist for each software delivery [Template (under the folder AWG Products / Developers Handbook / CURRENT)]. • MUST include install/uninstall package or script. • MUST include automated test drivers for all service operations in the CM. • For Web services, MUST include configuration files/scripts for CM startup and shutdown and sample consumer codes for invoking the services. • Deliver source codes and build scripts to the government if required.
4	<p>Version Description Document (VDD) [Template] (under the folder AWG Products / Developers Handbook / CURRENT)] [DI-IPSC-81442A]</p> <p>Delivery: same as that for Software End Items</p>	<ul style="list-style-type: none"> • Summarizes the features and contents of the CM, including a detailed list of Capability Packages and associated GOTS, COTS, and Operating System version numbers. • Identifies all changes made since the last VDD. • MUST include an inventory of documentation and software contents, installation instructions, known issues and workarounds. • MUST have a VDD for every unique delivery of the CM.
7	<p>Administration Manual (AM)</p> <p>Delivery: same as that for Software End Items</p>	<ul style="list-style-type: none"> • Describes the deployment of the Capability Packages. This includes importing from VMWare image files, configuration of internal network, disk storage, CPU, memory, etc. • Describes the configuration of the CM after it is deployed at GIG Computing Nodes. This includes license key setup, PKI certificate installation, key store setup, IP configuration, etc. • Provides detailed technical instructions and description of configuration files/scripts for CM startup and shutdown.
5	<p>Test Procedure and Report (TPR) [Template] (under the folder AWG Products / Developers Handbook / CURRENT)] [DI-IPSC-81440A]</p> <p>Delivery: same as that for Software End Items</p>	<ul style="list-style-type: none"> • Describes the procedure for testing the services under the CM (using the automated test drivers). Includes the expected results for each test. • The procedures MUST be based on the TEC Criteria and MUST provide clear indication of success or failure. • The report describes the results of formal testing done before delivery of the CM.
6	<p>Software Quality Assurance Report (SQAR) [DI-OCIC-81187]</p> <p>Delivery: same as that for Software End Items</p>	<ul style="list-style-type: none"> • A report by Software Quality Assurance personnel on the conformance of the CM to the requirements in the CM Specification. • It also verifies that the associated documentation conform to the requirements in the CM Work Package and this Handbook.

#	DELIVERABLES	DESCRIPTIONS
8	<p>Draft Service Level Agreement (SLA) [Template (under the folder AWG Products / Developers Handbook / CURRENT)]</p> <p>Delivery: 5 days after OTRR</p>	<ul style="list-style-type: none"> • Specifies the target metrics for the CM. • Includes Terms of Use, which identify the minimum hosting requirements and related parameters in order for the CM to fulfill the target metrics during operations. • The SLA Template may be tailored for a specific CM. • The Draft SLA will be used as input to the Hosting and Sustainment Work Package for the CM.
9	<p>DIACAP Package</p> <p>Delivery: 5 days before IARR</p>	<p>This package contains:</p> <ul style="list-style-type: none"> • IA registration for the CM in FDCE. • System Identification Profile (SIP), which documents the accreditation boundary, security features, architecture description, vulnerabilities, certification plans and test results, and supporting artifacts as the CM moves through the various FDCE stages. The SIP also serves as a security plan for the CM. • Existing Certification and Accreditation (C&A) documentation. If existing C&A documentation is not available, document the security plan for the CM using the NECC SIP template. • DIACAP Implementation Plan (DIP), which reflects the version / spiral targeted for evaluation. If existing documents are not available, develop them using the NECC template. • IA Self-Assessment Plan, which leverages the applicable security configuration guides and associated checklists, CM Test Team coordinated Detailed Test Plan. The plan identifies what automated tools are required (e.g. eEye Retina, Gold Disk, WebInspect, etc.), and describes the required testing support. • IA Plan of Action and Milestones (POA&M), which describes how the risks identified during the self assessment will be mitigated.
10	<p>Training Support Package (TSP)</p> <p>Delivery: 5 days after OTRR</p>	<ul style="list-style-type: none"> • This includes Programs of Instruction (POI), Lesson Plans (LP), Instructor/Student guide, and other instructional contents (DVD, CD, slides, etc.) currently existing in Joint and Service training institutions or centers. • This includes XML schemed documentation for EPSS distribution based on the GCCS-Maritime (GCCS-M) Document Management Infrastructure (DMI) guidance. • If required, the contents should be Electronic Performance Support System (EPSS) compliant. • If required, the contents should include Observe and Take Action Vignettes with SCORM® wrapper and SCORM® compliant courseware. • If applicable, include existing or new Computer Based Training (CBTs), and Web Based Training (WBT) packaged in SCORM compliant courseware packages and hosted on the Joint Knowledge Development and Distribution Capability (JKDDC) Learning Management System repository.
11	<p>Existing training materials</p> <p>Delivery: same as that for Training Support Package</p>	<ul style="list-style-type: none"> • Existing training materials applicable to NECC. Examples include user's manual, administrator guide, and other user/administration procedures in current form.

UNCLASSIFIED

#	DELIVERABLES	DESCRIPTIONS
12	CPE/UFP/OCE Documentation Delivery: [as required]	<ul style="list-style-type: none"> Provide documentation as defined in the CPE/UFP/OCE Standard Operating Procedure (SOP). Examples are: lessons learned, assessment, user feedback survey, etc.
13	Technology Readiness Level Evaluation Report Delivery: [as required] (1) draft due 5 days before PDR; (2) final due 5 days before CDR	<ul style="list-style-type: none"> Evaluates the Technology Readiness Level (TRL) of the technologies used by the CM. Uses the NECC Technology Readiness Assessment (TRA) as a basis for the evaluation.
14	Technology Maturity Plan (TMP) Delivery: same as that for Technology Readiness Level Evaluation Report	<ul style="list-style-type: none"> For each technology in the CM Technology Readiness Level Evaluation Report that evaluates at less than TRL 7, provide a Technology Maturity Plan as specified in the TRA Desk book.
15	Monthly Status Report [DI-MGMT-80227] Delivery: once a month beginning with start of contract	<ul style="list-style-type: none"> Cost and Schedule: Cost/Earned Value Management (to include development, program management, system engineering, hosting, piloting, and Hardware/Software) and Microsoft Project formatted schedule with major milestones and detailed dates for all deliverables plus the following: design, development, unit test, integration test, system test, accreditation, and CPE events. Risks: Identify risks, mitigation actions and plans Metrics: Report metrics data collected

APPENDIX D – INFORMATION FOR PROVISIONING CMS AT GCNS

Table 14 lists the information required for provisioning CMs at GCNs. Developers MUST complete the GCN pre-deployment checklist [Template (under the folder AWG Products / Developers Handbook / CURRENT)] and include it as part of each CM Software End Items delivery in Table 13.

Table 14: Information for Provisioning CMs at GCNs

ITEM	DESCRIPTION
Architecture diagrams	<ul style="list-style-type: none"> Logical architecture diagrams (may refer to diagrams in the SDD). Physical architecture (network) diagrams (may refer to diagrams in the SDD). These are typically site specific and should identify network interface cards.
Host names of Capability Packages (virtual machines)	<ul style="list-style-type: none"> List all the Capability Packages under the CM. Each Capability Package corresponds to a virtual machine. The virtual machine host names are assigned during configuration of the virtual machines.
Back-office components/ External services	<ul style="list-style-type: none"> List all back-office system dependencies and their host names. List all external service dependencies (e.g. NCES).
Domain Name Server entries	<ul style="list-style-type: none"> Specify Domain Name Server entries for all the Capability Packages and back-office components. Use fully qualified domain names (host name + domain name).
Computing resources required	<ul style="list-style-type: none"> If there are multiple Capability Packages, list the total number of CPUs, amount of memory, hard disk space, IP addresses, and server certificates needed. Identify the bandwidth requirement for local and wide area networks.
VMWare information	<ul style="list-style-type: none"> List the VMWare product used to create the virtual machine image files. List VMWare environment required at the GIG Computing Node (GCN), including: ESX Server version, Virtual infrastructure client, Virtual Center version.
Point of contact and documentation	<ul style="list-style-type: none"> List contact information for the CM. Identify documentation for the CM.
Information for each Capability Package / Back-office Component	
Server certificates	<ul style="list-style-type: none"> Indicate whether a server certificate is needed. Note: server certificates are typically provided by site and configured during installation.
Ports and protocols	<ul style="list-style-type: none"> Identify port exceptions required.
Processors	<ul style="list-style-type: none"> Specify the number of CPUs needed.
Memory	<ul style="list-style-type: none"> Specify the amount of memory needed in GB.

ITEM	DESCRIPTION
Hard disk space	<ul style="list-style-type: none">• Specify the amount of disk space needed in GB.• If multiple disks are used, each should be listed.
Software	<ul style="list-style-type: none">• Identify the guest operating system, COTS, GOTS, and their version / patch level.• Identify required software licenses.• Include those to be provided by the site, such as antivirus and vulnerability checker software.
Account information	<ul style="list-style-type: none">• Identify all required administrative and/or user accounts for the Capability Package / Back-office Component.

APPENDIX E – USAGE OF STANDARDS FOR NECC CAPABILITY MODULES

Table 15: Usage of Standards for NECC CMs

STANDARDS	CM USAGE GUIDANCE FOR DEVELOPERS
HTTP 1.1 TLS (IETF RFC 2246) X.509:2000	<ul style="list-style-type: none"> SHALL be used for transport and encryption all synchronous (request-response type) services
SOAP 1.2 WSDL 1.1 WS-I Basic Profile 1.1	<ul style="list-style-type: none"> SHALL be used for synchronous machine-to-machine services
WS-Security 1.1 SAML 2.0	<ul style="list-style-type: none"> SHALL be used for secured SOAP message exchange
CDSA 2.0 WS-Management (WS-Addressing, WS-Transfer, WS-Enumeration, XPath 1.0)	<ul style="list-style-type: none"> SHALL be used for synchronous machine-to-machine services built on the enterprise resource pattern SHALL support CRUD (Create, Retrieve, Update, Delete) and enumerated query operations
RSS 2.0	<ul style="list-style-type: none"> SHALL be used for lightweight event notification
REST-style Web Service	<ul style="list-style-type: none"> SHALL be used only for the following: <ul style="list-style-type: none"> user login to a Web server, read access to data for which any authorized user on the network is entitled to read, geospatial data and RSS services. For geospatial data services, developers MUST use the Open Geospatial Consortium standards (including WMS, SLD, WFS)
XML 1.1 (Second Edition) UTF-8 and UTF-16	<ul style="list-style-type: none"> SHALL be used for XML data
XML Schema Pt 1 and 2 XML Namespace	<ul style="list-style-type: none"> SHALL be used for schema definition
UCORE 1.0, DDMS 1.4.1 IC ISM v.2.0.3	<ul style="list-style-type: none"> SHALL import these common schemas in CM specific schemas
GML 3.2.1	<ul style="list-style-type: none"> SHALL import relevant parts of GML in CM specific schemas
HTML 4.01 XHTML 1.1 CSS2:1998 DOM 1.0	<ul style="list-style-type: none"> SHALL be used for presentation services SHOULD be used for AJAX type Web applications
JPEG PNG (ISO/IEC 15948)	<ul style="list-style-type: none"> SHALL be used for images in presentation services
MIL-STD-2525B(1)	<ul style="list-style-type: none"> SHALL be used for rendering tracks, tactical objects, etc. in presentation services
UDDI 3.0.2	<ul style="list-style-type: none"> SHALL be used with NCES Service Discovery Service

APPENDIX F – XML SCHEMA EXAMPLE

In this Appendix, a hypothetical CM called Target Manager (TGTM) is used to illustrate some of the conventions described in the main text. The Target Manager manages Target objects, which are associated with a TargetContext. Figure 3 shows the schema in graphic form for TargetContext, which is composed on a Target, a TrackingSource, and a UUID for the TargetContext.

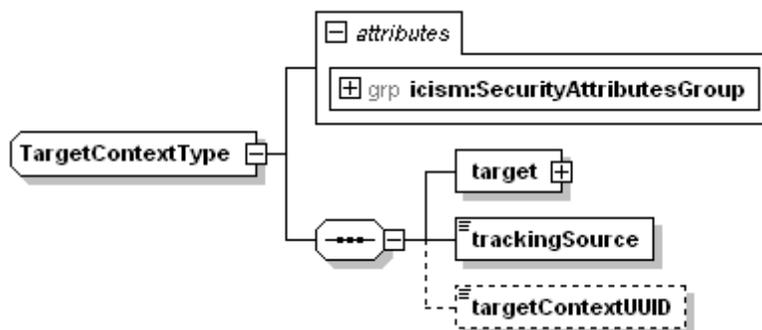


Figure 3: A Graphic View of the TargetContext Schema

Table 16 gives a partial view of the corresponding TargetContext XML schema, with the areas of interest highlighted. These areas are:

- Namespace format (line 2)
- Version number of the schema matching that in the namespace (line 8)
- Import of IC ISM schema (line 12) and its use as an attribute group (line 26)

Details of the related conventions are given in Section 3.2.

Table 16: A Partial View of the TargetContext Schema

```
(1) <?xml version="1.0" encoding="UTF-8"?>
(2) <xs:schema xmlns="http://metadata.dod.mil/mdr/ns/NECC-C2/FE-TGTM/1.0.0.0/
(3)   TargetContext" xmlns:xs="http://www.w3.org/2001/XMLSchema"
(4)   xmlns:t="http://metadata.dod.mil/mdr/ns/NECC-C2/FE-TGTM/1.0.0.0/Target"
(5)   xmlns:icism="urn:us:gov:ic:ism:v2"
(6)   targetNamespace="http://metadata.dod.mil/mdr/ns/NECC-C2/FE-TGTM/1.0.0.0/
(7)   TargetContext" elementFormDefault="qualified"
(8)   attributeFormDefault="unqualified" version="1.0.0.0">
(9)
(10)  <xs:import namespace="http://metadata.dod.mil/mdr/ns/NECC-C2/FE-
(11)    TGTM/1.0.0.0/Target" schemaLocation="TargetType.xsd"/>
(12)  <xs:import namespace="urn:us:gov:ic:ism:v2" schemaLocation="IC-ISM-v2.xsd"/>
(13)
(14)  <xs:complexType name="TargetContextType">
(15)    <xs:annotation>
(16)      <xs:documentation>
(17)        A TargetContext contains the target, the source for tracking the target,
(18)        and an optional UUID.
(19)      </xs:documentation>
(20)    </xs:annotation>
```

```
(21)     <xs:sequence>
(22)         <xs:element name="target" type="t:TargetType"/>
(23)         <xs:element name="trackingSource" type="SourceType"/>
(24)         <xs:element name="targetContextUUID" type="xs:string" minOccurs="0"/>
(25)     </xs:sequence>
(26)     <xs:attributeGroup ref="icism:SecurityAttributesGroup">
(27)         <xs:annotation>
(28)             <xs:documentation>
(29)                 A set of security markings that apply to this element.
(30)             </xs:documentation>
(31)         </xs:annotation>
(32)     </xs:attributeGroup>
(33) </xs:complexType>
...

```

APPENDIX G – AJAX APPLICATION EXAMPLES

Conventional Web application refreshes the whole page as it receives data synchronously. AJAX makes update on part of the page based on user interaction. Data communication may happen asynchronously in the background. This appendix provides some common examples of applying AJAX to user interfaces. Table 17 lists the common views described in Ref. 21 and examples of applying AJAX.

Table 17: AJAX Application Examples

VIEWS	EXAMPLES
Text - simple input field	<ul style="list-style-type: none"> As the user is typing, automatically suggest text values based on characters entered by the user
Text - drill down option	<ul style="list-style-type: none"> Refresh the next level of option based on the user selection at the current level. For example, after selecting a country, a country specific list of provinces/states is made visible.
Table	<ul style="list-style-type: none"> Allow interactive highlight of cells. Coordinate display of information on the page and cell selection. Asynchronous refresh of cell data.
Map	<ul style="list-style-type: none"> Pre-fetch neighboring map sections and show them as the user drags the pointing device. Show coordinated birds-eye view of a zoomed-in map. Allow users to select different map types. Allow users to select, customize, and enable overlays. Link to interactive information pane for selected overlays. Asynchronous refresh of overlays.
Chart	<ul style="list-style-type: none"> Allow interactive scaling of chart ranges. Show data point information interactively. Associate chart with table. Asynchronous refresh of chart data.
Alert	<ul style="list-style-type: none"> Fade-in/out of notification pane for new alerts.

By placing the views in Table 17 in multiple panes, one can construct rich AJAX user interfaces. Below are several such examples that can be applied to NECC presentation services.

Figure 4 is a multi-pane view with a form and a list. The pane on the right is a form for an object (a technical standard in this case), along with controls (search, add, update, etc.) for the object. The pane of the left has a list of objects as well as paging controls. The upper left corner may contain navigation links for the application. Users may enter values in the form fields and click “Search”. A list of objects is returned on the left. When the user selects one from the list, the form on the right is filled with values for the selected object. All these are done in AJAX style without refreshing the overall Web page. Such multi-pane views can be used in the presentation services associated with data service CMs.

The screenshot displays a web application interface with two main panes. The left pane, titled 'Navigation Links', contains a 'Standards' list with a search dropdown and a list of 20 items, including 'WSDL 1.1' which is selected. Below the list are pagination controls showing 'Page: 47 / 49'. The right pane, titled 'Standard', shows a form for the selected 'WSDL 1.1' object. The form includes fields for 'Identifier' (WSDL 1.1), 'Status' (Mandated), 'Version' (06-30), 'Service Area' (Document Interchange), and 'Owner' (Discovery TWG). It also has sections for 'Title', 'Summary', and 'Applicability', each containing descriptive text. At the bottom of the form, there are fields for 'Used In' (NECC-2006-11 NCES PTTA-v0.5 NCID), 'Comment' ([Add]), 'URL' (http://www.w3.org/TR/2001/NOTE-wsdl-20010315), and 'Dates' (Update: 2007.04.09, Created: 2006.07.03). A row of buttons at the bottom includes 'Search', 'Add', 'Update', 'Clear', and 'Delete'.

Figure 4: A Multi-pane View with a Form and a List

Figure 5 is another multi-pane view for a hypothetical Agent Management Tool. The pane with a list of Agent objects is to the right, on top of the Agent object pane. As before, both of these panes contain their own controls. An additional pane of the left contains groups of objects. Agent objects in the list pane can be dragged and dropped to a group on the left. Depending on the business rule, this may remove the dropped object from the list of Agents shown. Selection of an Agent group on the left will refresh the list of Agents on the list pane. Likewise, selection of an Agent will refresh the values in the Agent object pane.

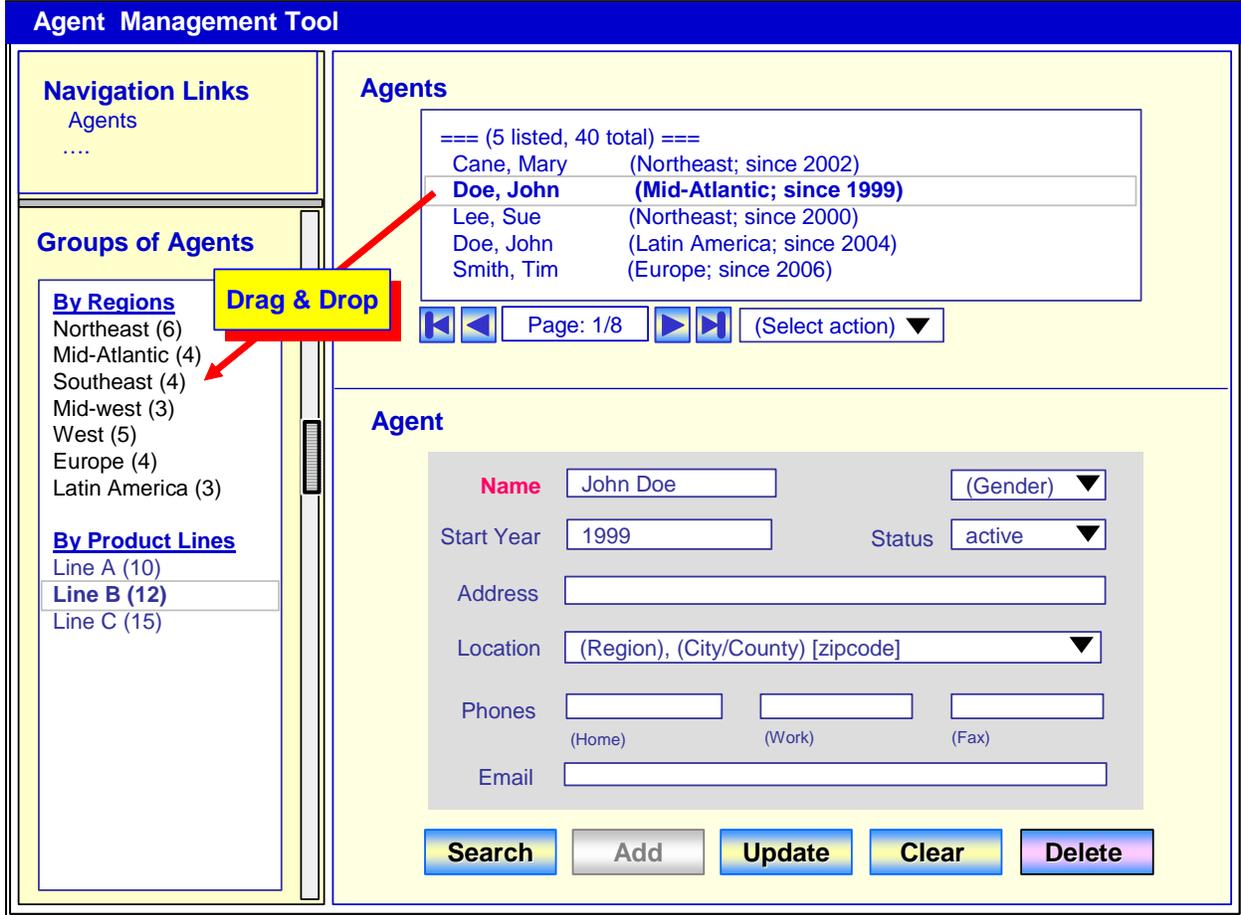


Figure 5: A Multiple Pane View with Groups of Objects in the List

Figure 6 is a map view from Google, which has the following features:

- The map pane itself contains all map-related controls (zoom, navigation, birds-eye view, map type, and overlay). Such a self-contained map makes the map service easily portable to different user interfaces. (Figure 6 has the traffic overlay turned on.)
- The list on the left pane and the markers on the map are mutually linked. Basic information about the entities (libraries in Figure 6) is contained in JavaScript data structures within the page. Selecting any one of them pops up an information view on the map. The map is also repositioned to center around the selected entity.

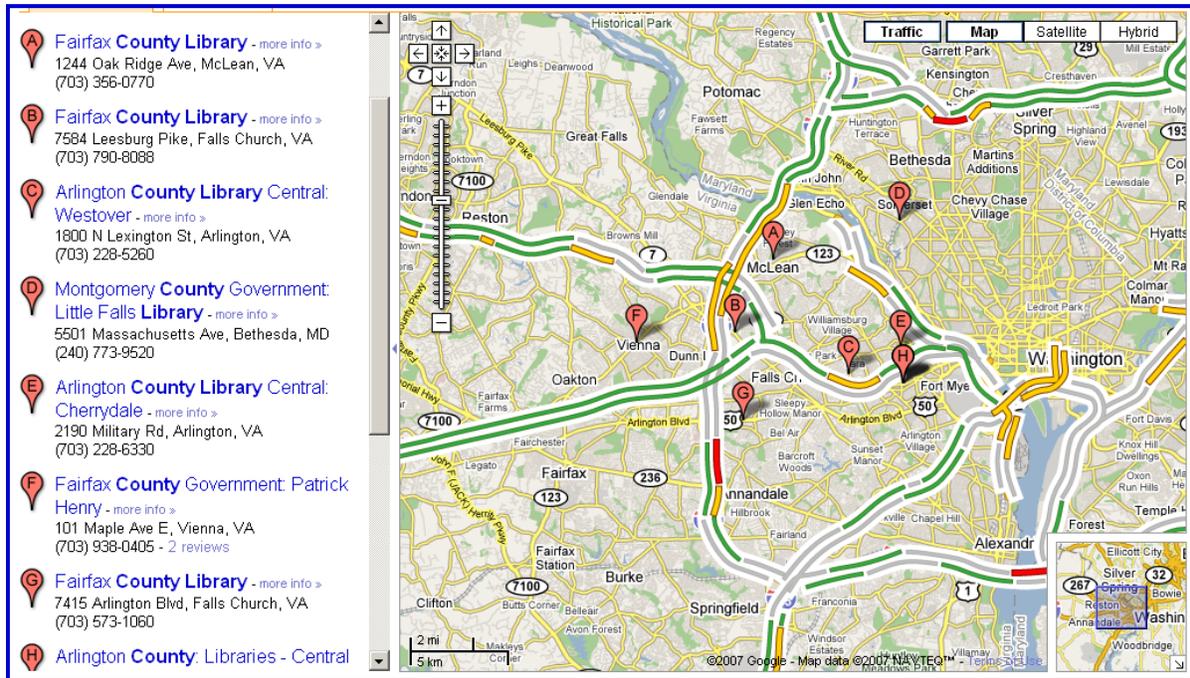


Figure 6: An AJAX-enabled Map View with a list pane on the left (from Google)

Figure 7 is a variation of Figure 6, with user selected tools. In this case, gas stations are shown as markers on the map. As the user drags the map, the markers are updated. Figure 6 also shows the path of a route highlighted by the user.

For NECC, these features can be applied to map-related services under Situational Awareness. Further, markers or other overlays on the map can be refreshed asynchronously to show current situation.

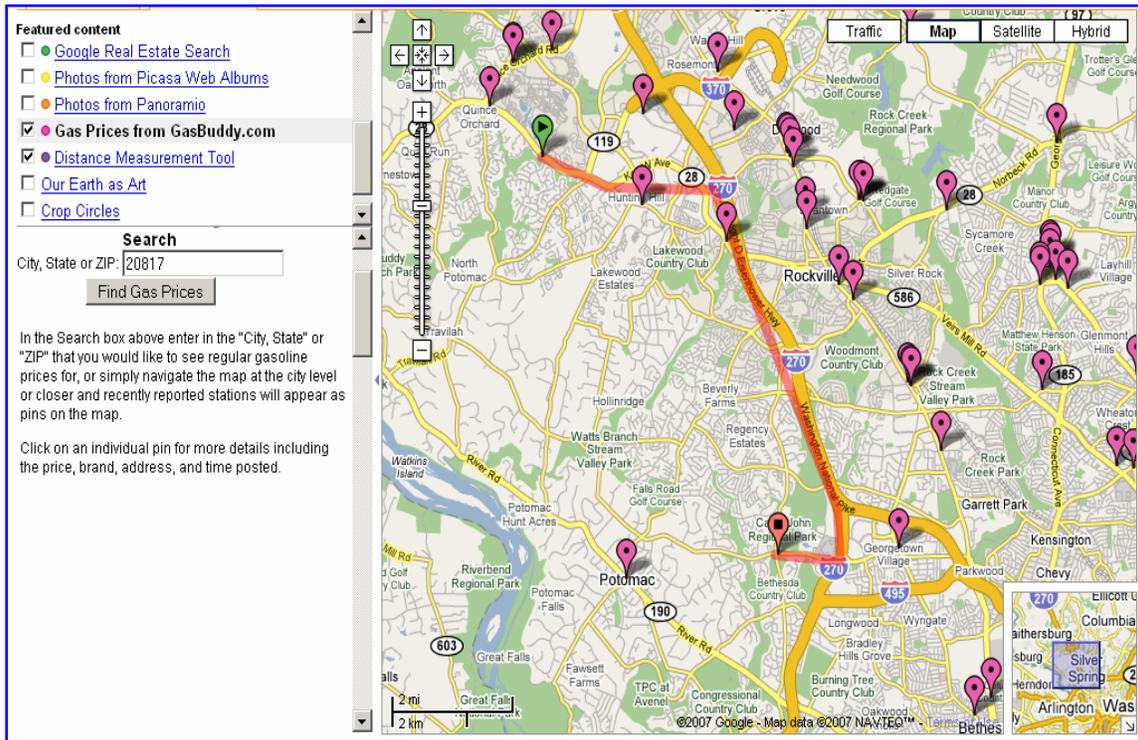


Figure 7: An AJAX-enabled Map View with user selected tools (from Google)

Similar multi-pane interfaces can be applied to other view combinations. Figure 5 is hypothetical Deployment Management Tool. The pane on the left is a tree view, with a list of units and corresponding Unit Line Numbers (ULNs). The pane on the right is a Gantt chart that shows the deployment schedule and milestones of the ULNs. Selecting of a ULN in one pane will highlight the same ULN in the other pane. Detailed information about the selected ULN, such as point of embarkation (POE) and point of debarkation (POD), may also be displayed for the selected ULN.

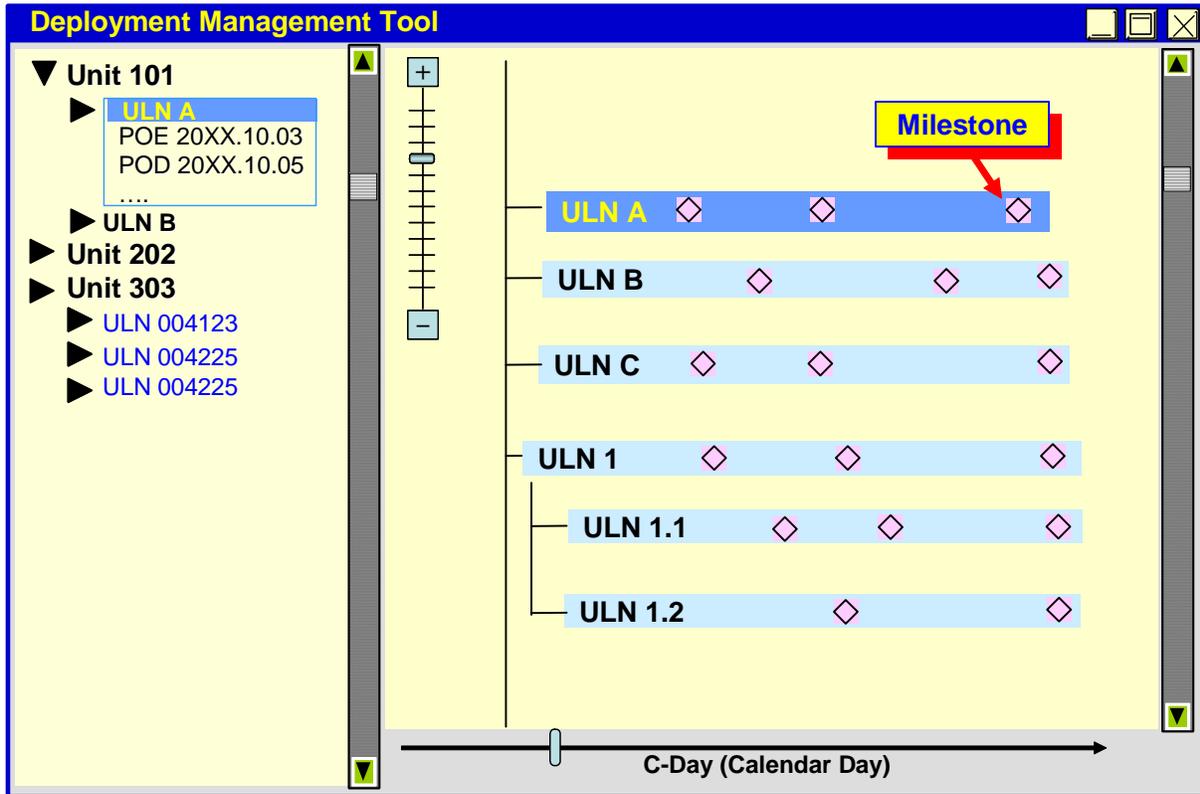


Figure 8: An AJAX-enabled Tree and Chart View

APPENDIX H – SCORM COMPLIANT COURSEWARE

This Appendix is applicable to developers who will produce training products. DoDI 1322.26 (June 2006) states that all DoD learning contents must adhere to the SCORM [Ref. 33]. This appendix provides quick reference information for developers who are required to develop SCORM Compliant Courseware.

SCORM is an XML-based standard that allows training modules developed by different authoring tools to interoperate with a LMS. Use of this standard allows learning contents to be launched, progress tracked, and results recorded through a central LMS. Contents compliant with SCORM are vendor neutral and can be moved from one LMS to another. SCORM 2004 3rd Edition is the most current version of the specification.

There are many SCORM compliant content authoring tools, ranging from simple to advanced levels. They generally offer graphical user interfaces and do not require programming or scripting. Many accept PowerPoint format as a starting point for creating the content.

For simple tools, it may only take one to two days to become proficient and one week of development for an hour of content. However, such tools are more restrictive in lay out and formatting. For advanced tools, it may take one month or more to become proficient and five to ten weeks of development for the same amount of content. These advanced tools provide much more features and allow precise lay out and formatting. Ref. 34 compares some of these SCORM authoring tools. Other tools are: ReadyGo Web Course Builder, IDL, iDesigner, and Microsoft LRN toolkit.

There are also many tools to convert existing training contents into SCORM compliant materials, making them ready for distributed learning systems. Some common and popular SCORM enabling tools are:

- ADLs – SCORM 2004 3rd Edition Conformance Test Suite
- Reload SCORM Editor

APPENDIX I – CONTACT INFORMATION

Table 18 lists the contacts for various NECC functional roles relevant to developers.

Table 18: NECC Contact Information for Developers

ROLE	NAME	CONTACT INFORMATION
Build Manager	Ammro Ragaban	ammro.ragaban@disa.mil
Configuration Management	Carolyn Stelter	stelerc@saic.com
Configuration Management	Jackie Harkins	harkinsj@saic.com
FDCE User Support	Michelle Smith	l.michelle.smith@saic.com
FDCE User Support	Carolyn Rivera	carolyn.rivera@disa.mil
NECC Help Desk	N/A	necchelp@navy.mil Phone: 800-838-1816 / 843-218-5665 / DSN: 588-5665 (use option 4)
NCES Help Desk	N/A	nces@csd.disa.mil Phone: 614-692-3136 / DSN 850-3136
NCES Service Registry Support	Robert DeVenny	robert.devenny@disa.mil
Piloting Support	Gary Vecchio	gary.l.vecchio@saic.com
Systems Engineering Review	Wallace Johnson	wallace.johnson@saic.com

APPENDIX J – ACRONYMS

Acronyms	Definitions
ABAC	Attribute Based Access Control
ADL	Advanced Distributed Learning
ADL-R	Advanced Distributed Learning-Registry
AJAX	Asynchronous JavaScript and XML
AM	Administration Manual
ASM	Association Management
ATEC	Army Test and Evaluation Command
C&A	Certification and Accreditation
C2	Command and Control
C2CS	Command and Control Common Services
CAC	Common Access Card
CBT	Computer-based training
CCB	Configuration Control Board
CDD	Capabilities Development Document
CDP	Capability Definition Package
CDR	Critical Design Review
CDSA	Context Data Source Adapter
CES	Core Enterprise Services
CF	Cross Function
CL	Confidence Level
CM	Capability Module
CMB	Configuration Management Board
CMTT	Capability Model Test Team
COI	Community of Interest
CONOPS	Concept of Operations
COTS	Commercial Off-The-Shelf
CPAS	Capability Provisioning Activities
CPE	Capability Provisioning Event
CPMO	Component Program Management Office
CPU	Central Processing Unit
CRUD	Create, Retrieve, Update, Delete
CSS	Cascading Style Sheet
DAA	Designated Approving Authority
DCID	Director of Central Intelligence Directive
DDMS	DoD Discovery Metadata Specification
DECC	Defense Enterprise Computer Center

Acronyms	Definitions
DIACAP	Defense Information Assurance Certification and Accreditation Process
DIL	Disconnected, Intermittent, or Limited
DIP	DIACAP Implementation Plan
DISA	Defense Information Systems Agency
DKO	Defense Knowledge Online
DMI	Documentation Management Infrastructure
DoD	Department of Defense
DoDAF	Department of Defense Architecture Framework
DoDI	DoD Instruction
DOM	Document Object Model
DSL	Definitive Service Library
DTRR	Developmental Test Readiness Review
ECA	External Certificate Authority
ECB	Early Capability Baseline
EMT	Engineering Mission Thread
EPSS	Electronic Performance Support System
ESB	Enterprise Service Bus
ESM	Enterprise Service Management
FDCE	Federated Development and Certification Environment
FE	Force Employment
FoS	Family of Systems
FPJ	Force Projection
FPT	Force Protection
FR	Force Readiness
FT	Functional Tutorial
GB	Gigabyte
GCCS	Global Command and Control System
GCN	GIG Computing Node
GES	GIG Enterprise Services
GES	Global Information Grid Enterprise Services
GIG	Global Information Grid
GML	Geography Markup Language
GOTS	Government Off-The-Shelf
GUI	Graphical User Interface
HCI	Human Computer Interface
HTML	Hyper Text Markup Language
I	Intelligence
I&TP	Integration and Technology Piloting
IA	Information Assurance

Acronyms	Definitions
IARR	Information Assurance Readiness Review
IATO	Interim Approval To Operate
IATT	Interim Approval To Test
IC	Intelligence Community
IDD	Interface Design Description
IMI	Interactive Multimedia Instruction
IOT&E	Initial Operational Test and Evaluation
IP	Internet Protocol
ISM	Information Security Marking
ISO	International Standards Organization
JCCD	Joint Combat Capability Developer
JFCOM	Joint Forces Command
JFS	Joint Force Synchronization
JITC	Joint Interoperability Test Command
JKDDC	Joint Knowledge Development and Distribution Capability
JPMO	Joint Program Management Office
JST	Joint System Team
JTOCC	Joint Technical Operations Control Capability
LMS	Learning Management System
LRA	Local Registration Authority
M2M	Machine-to-Machine
MAC	Mission Assurance Category
MDR	Metadata Repository
METL	Mission Essential Task List
MOA	Memorandum of Agreement
MSG	Messaging
NCES	Net-Centric Enterprise Services
NECC	Net-Enabled Command Capability
NIPRNet	Unclassified but Sensitive Internet Protocol Network
NIST	National Institute of Standards and Technology
NR-KPP	Net-Ready Key Performance Parameters
NSA	National Security Agency
OCE	Operational Concept Experiments/Events
OGC	Open Geospatial Consortium
OMT	Operational Mission Threads
OS	Operating System
OTRR	Operational Test Readiness Review
PDR	Preliminary Design Review
PKI	Public Key Infrastructure

Acronyms	Definitions
PL	Protection Level
POA&M	Plan of Actions and Milestones
POC	Point of Contact
POD	Point of Debarkation
POE	Point of Embarkation
RA	Registration Authority
RAA	Readiness Assessment and Analysis
RBAC	Role Based Access Control
REST	Representational State Transfer
RFD	Request for Deviation
RSS	Really Simple Syndication
SA	Situational Awareness
SCI	Sensitive Compartmented Information
SCORM	Shareable Content Object Reference Model
SDD	Software Design Description
SDK	Service Development Kit
SE	Systems Engineering
SEP	Systems Engineering Plan
SIP	System Identification Profile
SIPRNet	Secure Internet Protocol Routed Network
SLA	Service Level Agreement
SLD	Styled Layer Descriptor
SOA	Service Oriented Architecture
SOAF	Service Oriented Architecture Foundation
SOAP	Simple Object Access Protocol
SOP	Standard Operating Procedure
SORTS	Status of Resources and Training System
SPS	System Performance Specification
SQAR	Software Quality Assurance Report
SS	Service Security
SSL	Secure Sockets Layer
SSTM	Single Service Training Manager
STIG	Security Technical Implementation Guide
T&E	Test and Evaluation
TEC	Test, Evaluation, and Certification
TEMP	Test and Evaluation Master Plan
TGTM	Target Manager
TLS	Transport Layer Security
TRA	Technology Readiness Assessment

Acronyms	Definitions
TRL	Technology Readiness Level
UCORE	Universal Core
UDDI	Universal Description, Discovery, and Integration
UFP	User Free Play
ULN	Unit Line Number
UML	Unified Modeling Language
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTC	Coordinated Universal Time
UUID	Universal Unique ID
VDD	Version Description Document
WAR	Web Application aRchive
WBT	Web-based Training
WCS	Web Coverage Service
WFS	Web Feature Service
WMS	Web Map Service
WS	Web Service
WSDL	Web Service Description Language
XHTML	Extensible Hypertext Markup Language
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformation