NPS-AM-08-034

EXCERPT FROM THE

PROCEEDINGS

OF THE

FIFTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM

SHARE REPOSITORY FRAMEWORK: COMPONENT SPECIFICATION AND OTOLOGY

Published: 23 April 2008

by

Jean Johnson and Curtis Blais

5th Annual Acquisition Research Symposium of the Naval Postgraduate School:

Acquisition Research: Creating Synergy for Informed Change

May 14-15, 2008

Prepared for: Naval Postgraduate School, Monterey, California 93943

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

The research presented at the symposium was supported by the Acquisition Chair of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

To request Defense Acquisition Research or to become a research sponsor, please contact:

NPS Acquisition Research Program
Attn: James B. Greene, RADM, USN, (Ret)
Acquisition Chair
Graduate School of Business and Public Policy
Naval Postgraduate School
555 Dyer Road, Room 332
Monterey, CA 93943-5103
Tel: (831) 656-2092
Fax: (831) 656-2253
E-mail: jbgreene@nps.edu

Copies of the Acquisition Sponsored Research Reports may be printed from our website www.acquisitionresearch.org

Conference Website:
www.researchsymposium.org

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

Proceedings of the Annual Acquisition Research Program

The following article is taken as an excerpt from the proceedings of the annual Acquisition Research Program. This annual event showcases the research projects funded through the Acquisition Research Program at the Graduate School of Business and Public Policy at the Naval Postgraduate School. Featuring keynote speakers, plenary panels, multiple panel sessions, a student research poster show and social events, the Annual Acquisition Research Symposium offers a candid environment where high-ranking Department of Defense (DoD) officials, industry officials, accomplished faculty and military students are encouraged to collaborate on finding applicable solutions to the challenges facing acquisition policies and processes within the DoD today. By jointly and publicly questioning the norms of industry and academia, the resulting research benefits from myriad perspectives and collaborations which can identify better solutions and practices in acquisition, contract, financial, logistics and program management.

For further information regarding the Acquisition Research Program, electronic copies of additional research, or to learn more about becoming a sponsor, please visit our program website at:

www.acquistionresearch.org

For further information on or to register for the next Acquisition Research Symposium during the third week of May, please visit our conference website at:

www.researchsymposium.org

THIS PAGE INTENTIONALLY LEFT BLANK

SHARE Repository Framework: Component Specification and Ontology

Presenter: Jean Johnson is a research assistant in the Naval Postgraduate School Systems Engineering Department. She coordinates a program to develop Modeling and Simulation curriculum for DoD Acquisition workforce personnel and performs research for the PEO IWS SHARE program. Previously, Johnson held various positions supporting NAVSEA's Warfare Systems Engineering Division. She is a US Navy active and reserve veteran and continues her Navy affiliation in the Individual Ready Reserve. Johnson holds a ME in Operations Research and Systems Analysis and a BS in Applied Mathematics from Old Dominion University and is currently pursuing her PhD in Software Engineering at NPS.

Jean Johnson
Naval Postgraduate School
1 University Circle
Monterey, California 93943
E-mail: jmjohnso@nps.edu
831-656-2956

Author: Curtis Blais is a research associate in the Naval Postgraduate School Modeling, Virtual Environments, and Simulation (MOVES) Institute. His research interests include application of Web-based technologies to improve interoperability of C2 systems and M&S systems. He is contributing to metadata design efforts for the SHARE program, the Department of Defense (DoD) M&S Community of Interest Discovery Metadata Specification, M&S Catalog, and standardized DoD Verification, Validation, and Accreditation (VV&A) documentation, as well as international standardization efforts for the Military Scenario Definition Language and the Coalition Battle Management Language. Blais hold BS and MS degrees in Mathematics from the University of Notre Dame and is a PhD candidate in the MOVES program.

Curtis Blais
Naval Postgraduate School
1 University Circle
Monterey, California 93943
E-mail: clblais@nps.edu
831-656-3215

Abstract

Data sharing is the information technology watchword of our time. Revolutions in information exchange and interoperability are underway in government and industry through policies on the strategic end to data standards on the implementation end. The revolution is transforming acquisition systems and processes through specification of open architectures, which enables construction of new complex systems from crafted components. In August 2006, Program Executive Officer, Integrated Warfare Systems (PEO IWS), established the Software, Hardware Asset Reuse Enterprise (SHARE) repository to enable the reuse of combat system software and related assets. The Naval Postgraduate School (NPS) is tasked to develop a component specification and ontology for the SHARE repository framework. This paper summarizes the work completed to date in support of the PEO IWS 7-sponsored NPS research project "Software Component Specification Framework and Ontology for the SHARE Repository." We describe SHARE and the vision for the repository framework. We present

related technologies and the next steps for the framework development. Finally, we provide ideas for future research.

Introduction

Typical challenges for reuse repositories include the lack of motivation for reusable component developers and those wishing to reuse the components, the difficulty of component retrieval and selection, and the amount of work necessary to integrate a selected component. Often, a developer seeking reusable assets from a repository has difficulty finding the desired items due to the inability of the repository designers to foresee what will constitute valuable criteria to guide the search. As a result, the metadata in the repository search tool does not provide adequate information to enable efficient decisions about which assets to pursue. Additionally, most common searches are conducted based on key words and phrases; this requires that the searcher knows how to express desired component features in the terminology employed by the submitter of the asset or the repository manager and its developers.

To further complicate the matter, the goals of reuse depend on the activity being performed within the software lifecycle. To enable reuse during each activity, the desired assets will differ as well as the information required about those assets. For example, during the requirements activities, the sought-after reusable assets may be requirements for similar systems or design artifacts; while at implementation time, developers searching a database may be seeking bits of code or interface specification information. In order for a single repository to support these different searches, the repository builders must incorporate sufficient metadata information to support each type of search. With this perspective, it is clear that the provision of inclusive metadata for components in a database is difficult at best.

In August 2006, the Program Executive Officer, Integrated Warfare Systems (PEO IWS) established the Software, Hardware Asset Reuse Enterprise (SHARE), a library of combat system software and related assets for use by eligible contractors (both prime contractors and subcontractors) for developing or suggesting improvements to Navy Surface Warfare Systems. SHARE is one piece of the Navy's Open Architecture (OA) approach to developing modular, open systems (PEO IWS, 2007), which includes reusable software applications as a core principle. PEO IWS is currently seeking ways to improve and mature the capability provided by SHARE. Among other initiatives, two related research projects are in progress at the Naval Postgraduate School (NPS). The first, and the topic of this paper, will produce a component specification and ontology for use in SHARE. The second will develop a prototype of a semantically based requirements search engine (ReSEARCH) with the tools necessary to convert documents into semantically based formal representations of requirements (Martel, 2007).

The component specification will describe the artifacts contained in the repository in sufficient detail to aid a repository user in determining whether the artifact is worth retrieving. The ontology will provide contextual semantics describing relationships among items in the repository to aid in associating artifacts with user needs. The component specification and ontology will comprise a rich structural and semantic framework for SHARE that will enable multiple kinds of search and discovery techniques. The goal is to enable the development of different types of tools to improve the usefulness of SHARE.

In this paper, we present the work completed to date on the SHARE ontology and component specification project. We describe the SHARE repository, present our conceptual

vision for the repository framework, and describe the technologies that will be used in its development. We also summarize related work and our plans for completion of the framework.

SHARE Repository

SHARE provides a capability for discovering, accessing, sharing, managing, and sustaining reusable assets for the Navy Surface Domain's programs (Belcher, 2007)[1]. It consists of an asset library and a card catalog. The asset library is a collection of combat systems software and supporting artifacts. As of January 2008, 62 assets containing 18,017 artifacts from the Aegis, Ship Self Defense System (SSDS), Littoral Combat Ship (LCS), DDG-1000, and Single Integrated Air Picture (SIAP) programs were available in the SHARE library. The card catalog is a Web-based interface that facilitates user insight into the contents of SHARE and supports user functions—including account registry, asset search and discovery, asset submission assistance, and asset retrieval requests.

The SHARE asset library is separate from the card catalog for two primary reasons. First, the majority of the contents of SHARE is classified material and, therefore, must be kept in a SECRET or higher container. Second, the process for retrieving assets from SHARE includes necessary steps for addressing the data rights associated with each component. For most of the components, a license agreement and Non-Disclosure Agreement are required before an asset can be issued (these forms are also available on the website). Due to these restrictions, the Web interface and the actual assets are physically separated.

The search and discovery process in SHARE is conducted either through individual navigation of the list of assets in the catalog or by a keyword search of the metadata contained in the catalog. From the catalog list, a user can select an asset to obtain a more detailed description, consisting of identity, description and usage information if available.

Assets are requested from SHARE using an online interactive questionnaire. The tool then prepares the necessary documents (including non-disclosure and license agreements) and provides them, along with instructions for printing and submission, to the user. Once the documents have been mailed to the SHARE administrators, the user can track the status of the request online through the SHARE interface. When approved for distribution, library materials are provided either online through access to the appropriate portion of the SHARE website (classified or unclassified) or via delivery of physical media.

A more complete description of SHARE is available in Johnson (2007).

Conceptual Vision for the Software Repository Framework

Popular software repositories, such as SourceForge (2007) and CPAN (2007), tend to be organized to support keyword searches over broad categories of software types. Essentially two types of search are enabled. Items in the repository are grouped by type or function and are then browsable within those categories. A keyword search over metadata is also possible. Repository entries vary greatly in the amount of information (metadata) available for each registered artifact.

[1] Organizations interested in registering for access to the library should visit and complete an online registration form at https://viewnet.nswc.navy.mil.

The Goal: Improved Search Capability

Current repositories do not support all the types of search we would like to enable. In addition to typical types of search, we envision a graphical user interface that enables navigation of repository assets depending on user's interests. This requires an interface that allows users to project their context on the search mechanisms. In other words, users bring particular information needs and goals based on the problem they are trying to solve. The interface needs to have natural mechanisms to enable users to pose inquiries that fit readily with their views of the problem space. For example, users may be seeking particular functionality best obtained through a functional view or by sorting the information in the repository. Or, users may be seeking particular artifacts best obtained through a document resource organization of the information. Or, users may be seeking information on certain testing methodologies that have been applied so that organization of the information by work activity would best apply. The challenge in designing the framework for the software repository is devising initial sets of such taxonomic descriptions of the assets while creating flexibility for future introduction of additional and diverse organizational views (profiles or templates) of the information as user needs and repository utility grow.

Fish-eye Graph

One example of the type of tool that could be supported by the framework is a fish-eye graph (Sarkar & Brown, 1993). This is a visualization tool that has not, to date, been used to aid in navigation of repository contents. Fish-eye graphs, depicted in Figure 1. display to the user objects of interest in addition to the relationships that the objects have with other items. As the relationships of interest to the user are explored, the graph highlights the item and brings it to the front of the display. The user can then weed out uninteresting items by removing from view the relationships that are not important. This type of search results in a single or small grouping of items that the user has found interesting, with supporting information available by mouse-click.

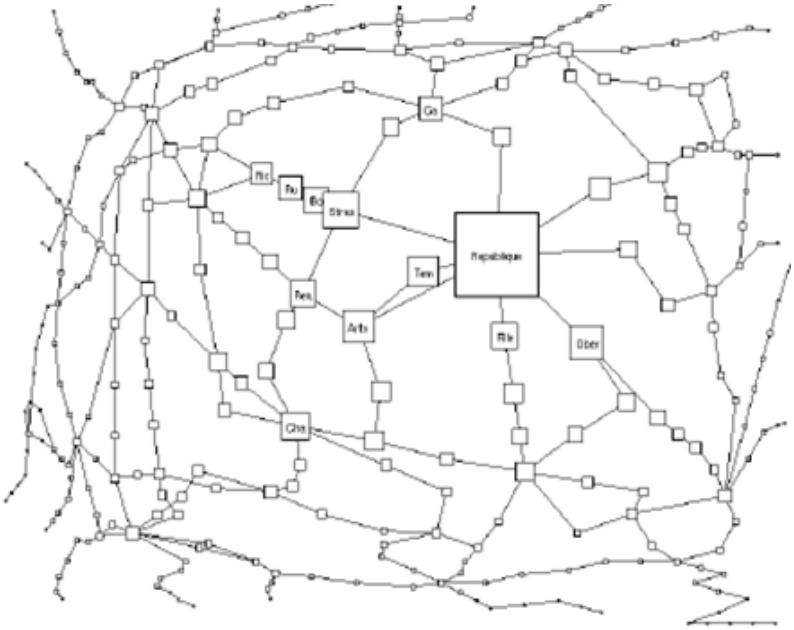

Figure 1. Example Fish-eye Graph
(Sarkar & Brown, 1993)

Semantic Search

Current repository metadata schemas do not address issues of language ambiguity. Rather, they assume that the keywords provided by the metadata will match identically to the words inserted by the user. By providing a framework of related concepts in which to place the artifacts, a search tool can be designed to navigate for artifacts in such a way that users do not need to know the initial set of exact words used to describe the artifacts.

A related ongoing NPS research project titled ReSEARCH is focused on solving these types of issues for SHARE. This work intends to enhance current search mechanisms, principally Latent Semantic Indexing (LSI), by employing word-sense relationships provided in the extensive WordNet lexical database (Princeton, 2006). However, this body of work lacks the domain-specific lexicon found in focused endeavors, such as Navy combat systems. Formalized semantic descriptions in the SHARE component specification and ontology will further enhance ReSEARCH capabilities to produce highly relevant search findings for users of the SHARE repository.

Model-based Search

A third type of search we have envisioned is based on a user-constructed model of the problem the user is trying to solve. The user interface for the repository can provide the capability to assist the user in building the model of a desired system architecture using a standardized representation scheme (e.g., Unified Modeling Language), and the search can then return possible existing solutions for portions of the system and demonstrate potential gaps. Model-based search has similarities to the semantic search concept described above—taxonomic and ontological descriptions of systems, system components, lifecycle phases, development artifacts, usage, and other concepts prominent in the software-hardware domain of SHARE provide structural information that can greatly facilitate search of available assets.

The metadata collected in current repositories do not support these types of advanced discovery tools. Here we provide an overview of our approach for developing a repository that will enable these types of search capabilities.

The Solution: The SHARE Framework

To enable the types of tools we envision, we must create a richer semantic framework for the repository. The framework will be composed of two parts: the component specification and the ontology.

Component Specification

The component specification is a description or model of the items in the repository. For our efforts, we will focus on two aspects of the component specification: “typical” metadata and software behavior.

Metadata—The metadata for each artifact should incorporate all necessary data for discovery and implementation. The metadata will both aid repository users in determining if the item is suited for their use, and provide information about how to use the retrieved asset. We refer to this as “standard” or “typical” metadata since there are many existing examples of metadata that we can use to develop such metadata for SHARE.

Software Behavior—The metadata for many current repositories, such as those described earlier, fail to capture a searchable representation of the functionality of the items outside

general categories of functionality (e.g., Archiving Compression Conversion, Control Flow Utilities, Graphics, Security) and text-based search. Unlike current practice, the SHARE component specification will consist of both typical metadata and a behavioral model of the component. Since this piece of the component specification is not commonly incorporated into repositories in a standardized manner, we feel it is a specific focus area to identify the appropriate representation mechanisms for software behavior in the repository context.

Ontology

The second part of the framework is description of the relationships of the components. These form a contextual model of the repository items to represent a particular perspective that can more closely match a user's problem context. These relationships may include the component's use/role in existing systems, its mapping to reference or domain architectures, its utility in various software development lifecycle phases, and other types of relationships we expect to discover during the research. Consider the example relationships among artifacts shown in Figure 2. Suppose we insert a requirements document for a particular component into the repository. This artifact may have been originally developed for System A in the figure. The item's relation to the rest of the original system provides the context for one dimension of the repository framework. If this item was then reused to fulfill some requirements of System B, its location in that model provides a second dimension. Additionally, the requirements document will map to some taxonomy of artifacts that are relevant for particular phases of the product lifecycle. Finally, the component it describes may also have a place in some domain-specific reference architecture. All these relationships provide contextual information about the artifact that can be exploited to enable sophisticated search and discovery methods described above. For this project, an appropriate representation of component context will be identified and the relationships defined. This will enable navigation of the repository based on the contextual information provided in the ontology.

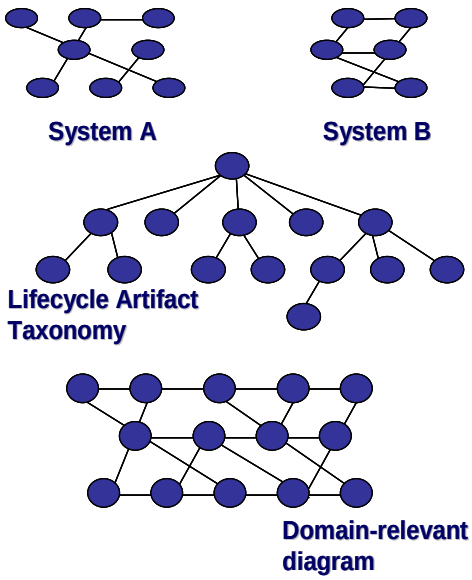

Figure 2. Artifact Relationships

Based on this vision then, the project team has identified three focus areas for developing the framework for the SHARE repository:

1. “Typical” metadata for artifacts
2. A suitable representation of software behavior
3. Framework relationships (ontology)

The current research project will focus on building each of these items for the SHARE repository. Follow-on work will be required to implement the framework in a tool suite that will enable the search capabilities described above. This and other suggested follow-on work is described in the future work section of this report.

Related Technologies

There have been concerted efforts in recent years to add semantics to stored information in order to promote a greater ability to support automated search and sharing of information. The following subsections highlight a number of efforts that can inform our design of the SHARE repository framework.

Metadata Initiatives

There has been much work done on specification of metadata to describe assets and resources in various repositories. For the SHARE framework, we do not expect to create any unique approaches to developing metadata, nor will we develop any fundamentally different metadata set. However, we intend to use the metadata descriptions to support navigation-by-context search, in addition to performing more traditional types of searches based on keywords, text-analysis, and popularity.

World Wide Web

The World Wide Web has experienced unprecedented growth over the past 20 years. This growth was largely fueled by the use of Hypertext Markup Language (HTML), Hypertext Transfer Protocol (HTTP), and Uniform Resource Identifiers (URIs) as simplistic mechanisms for putting information into document files, posting and accessing those files, and linking across those files, respectively. However, HTML primarily describes how the information should be displayed in browser software, rather than providing clear descriptions of the information content of the document. To address this shortcoming, the World Wide Web Consortium (W3C) created the Extensible Markup Language (XML) as a standard way to create and apply markup to the content of Web documents to make the content more readily accessible by software. While initial application of XML made description of Web content much more precise, it largely described content in a structured, syntactic manner. As the demand for greater automation in accessing and processing Web content continued to rise, principal designers and researchers on the Web created a new vision called the Semantic Web.

The Semantic Web is Tim Berners-Lee’s vision of the World Wide Web (Berners-Lee, Hendler & Lassila, 2001) in which vast stores of information become meaningful to computers and where “the explicit representation of the semantics underlying data, programs, pages, and other Web resources will enable a knowledge-based Web that provides a qualitatively new level of service” (Daconta, Obrst & Smith, 2003, p. xxi). The Semantic Web is an extension of the World Wide Web in which information is given semantically rich descriptions that enable automated processing by software. The W3C has created additional layers of markup, building on the base of XML, to provide description of the semantics of the information. The Semantic Web is an evolution of the current Web, built from the foundation of open standards on which the Web is built. Building blocks of the Semantic Web are shown in Figure 3. Below, we provide

a brief description of the base layers of the Semantic Web stack (URI/IRI and XML) and highlight their relevance to the SHARE metadata development. Applicability of other components of the Semantic Web stack to the SHARE framework will be discussed later in this paper.

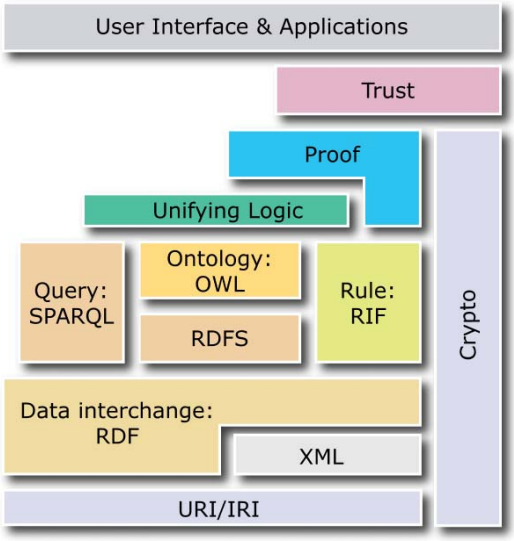

Figure 3. Principal building blocks of the Semantic Web Stack
(W3C, 1994)

Data Sharing Policies in the US Government

In the US Department of Defense, including DoD intelligence agencies and functions, the guiding document for information sharing is the Net-Centric Data Sharing Strategy (DoD Chief Information Officer, 2003). The document defines net-centricity as "the realization of a networked environment, including infrastructure, systems, processes, and people that enables a completely different approach to warfighting and business operations" (DoD Chief Information Officer, 2003, p. 1). The network foundation is the Global Information Grid, "the globally interconnected, end-to-end set of information capabilities, associated processes, and personnel for collecting, processing, storing, disseminating, and managing information on demand to warfighters, defense policymakers, and support personnel" (DoD Chief Information Officer, 2003, p. 1). Data assets addressed by the strategy include system files, databases, documents, official electronic records, images, audio files, websites, and data access services. Users and applications can search for and "pull" data as needed, or they can receive alerts when their subscribed data is updated or changed (publish/subscribe). The goals of the strategy are to make data

- visible—users and applications can discovery the data assets
- accessible—users and applications can obtain the data assets
- institutionalized—data approaches are incorporated into DoD process and practices
- understandable—users and applications can comprehend the data, both structurally and semantically to address specific needs

- trusted—users and applications can determine the authority of the source of the data assets
- interoperable—metadata is available to allow mediation or translation of data to support many-to-many exchanges of data
- responsive to user needs—mechanisms for improvement through continual feedback are supported to address particular perspectives of data users. (DoD Chief Information Officer, 2003, p. 10)

The design of the repository framework should provide or support mechanisms that address each of these goals. In this respect, the data sharing goals help to guide the design and development efforts. A guiding document is the DoD Discovery Metadata Specification (DDMS) (Deputy Assistant Secretary of Defense, 2007), which provides a standard set of metadata for discovering distributed resources. The SHARE repository, as with other repository efforts, can readily address the DDMS by ensuring that sufficient metadata are provided in descriptions of assets to allow generation of at least the minimum required set of metadata specified in the DDMS.

Existing Repository Metadata

Outside the DoD, there are additional metadata practices from which we can learn. All existing repositories have some sort of metadata schema, whether well defined or not. Also, there are some specific efforts that focus on the development of metadata standards for use in software repositories. Some examples of each of these are discussed here.

As introduced earlier, open source repositories such as SourceForge and CPAN have a metadata set for describing their contained assets. Unfortunately, these schemas are not often published. However, they can be somewhat derived simply by looking at the available information for each of the items in the repository.

Another approach is the Object Management Group (OMG) Reusable Asset Specification (RAS) standard for the packaging of software assets. The RAS describes required and optional classes, as well as required and optional attributes, for packaging software assets. The specification is depicted as Universal Modeling Language (UML) models, which are translated into XML Schema and Meta-Object Facility (MOF) / Extensible Metadata Interchange (XMI) XML Schema. In the RAS, artifacts are defined as “any work products from the software development lifecycle,” and assets are a grouping of artifacts that “provide a solution to a problem for a given context” (Object Management Group, 2005, p. 7). Accordingly, the RAS describes an approach for packaging artifacts into an asset using a manifest file.

For SHARE metadata development, we will use these existing examples of metadata as references when the metadata schema is developed. Existing metadata sets will be used to trigger the evaluation of items that could be included but were not originally considered. The goal is not to merge all existing sets of metadata but to assess the relevance of existing data sets for SHARE and include any appropriate items.

Software Behavior Representation

Repositories today tend to capture software behavior as key words describing a general functional area or as a free text description field in the metadata. This type of description can be helpful to users in determining whether the item will be useful in meeting their needs. However, if the desired end-goal is more sophisticated than today’s repository capabilities, a more formal description of behavior is required. For example, one of the loftier goals of a

software repository may be to automatically compose systems from reusable components. This is a difficult problem, which many have tried to solve[2]. It is especially difficult if the components were not originally designed for reuse. As a necessary first step towards more sophisticated uses of a repository, behavioral descriptions must be machine-readable in order to support automated search and discovery. Furthermore, the behavior descriptions must be formalized and consistently applied to each item in the repository if the intent is to automatically compose them into a larger functioning system.

A formalized description of software behavior typically means one of two things. We either (1) define the inputs and outputs (interfaces) of the components, or we (2) describe the operations that take place within the component. Many people view the latter as a decomposition of the former. In other words, they describe the inner workings of a component by defining the inputs and outputs of a more granular subset of components. Therefore, here we summarize the current approaches for documenting both software interfaces as well as behavior.

Interface Descriptions

Interface descriptions focus on the inputs and outputs of a component and not the inner workings of that component. Interfaces are represented using various methods, which vary from specific concentration on the connect points between two pieces of software and the types of information passed across them, to representations of the services that a component provides.

One often-employed method for representing interfaces in component software technologies is as a contract between the client and the provider of the implementation (Szyperski, 2002, p. 53). The contract defines the services promised by the interface and the requirements of the client for using the interface. It could simply consist of a set of named operations that can be invoked by clients. It may also include pre- and post-conditions necessary for the successful use of the interface. A drawback for using this type of interface description as a basis for search and discovery in SHARE is the dependency on the component's originating software language for determining the syntax and semantics used to describe the operations and conditions. In SHARE's heterogeneous environment, these types of standardized descriptions may not be practical.

Component technology developers have developed Interface Definition Languages (IDLs) to specify interfaces independently of the programming language used for source code development (Clements et al., 2002, p. 554). Examples include OMG IDL and Microsoft's COM IDL. The same drawback discussed for the programming language-dependent contracts for our heterogeneous SHARE environment exists for these intermediate languages. Rather than a dependency on the programming language, however, the dependence here lies in the chosen component technology. Since we do not intend to force a specific component technology for all SHARE contributors, it does not make sense to insist on interface definitions based on these IDLs.

[2] The proceedings from the International Symposium on Software Composition, an annual event, provide examples of research into the breadth of research topics currently being pursued in the area of software composition. The website for the 2008 conference is located at http://www.2008.software-composition.org/.

Architecture Description Languages (ADLs) are primarily used to formally represent system architectures for use during development and typically describe system elements, their interactions, and their composition rules. While there are many different viewpoints about what constitutes an ADL (Medvidovic & Taylor, 2000, pp. 71-72), they always include a formal description of interfaces. ADL interface descriptions typically define the required and provided services (messages, operations, and variables) of a component. Some ADLs also allow for parameterization of interfaces; others provide additional information.

The advantage of using ADLs in a component specification is that the benefits of ADL-based tools may be realized for the components. ADL tools assist the developer by supporting architecture creation, visualization, validation, refinement, simulation, and analysis, in addition to features that enable systematic transformation of architectures into the implementation of a system. Many support generation of "glue code" for components once their implementations are developed. Additionally, ADLs are likely the appropriate level of abstraction for a heterogeneous collection of assets, such as those found in SHARE, since they do not depend on any decisions made about the implementation of the components.

Unfortunately, the use of ADL-type descriptions does come with a cost. Because of the robust descriptive capabilities of many ADLs, there is considerable effort required in learning how to use them. This would present a learning curve for both asset submitters and retrievers. To minimize this problem, tools could be developed to aid the user in producing the required ADL descriptions. As an alternate solution, we are investigating the possibility of incorporating some ADL-like descriptions into the XML-defined metadata for the components. This will enable us to incorporate only those aspects that are relevant to the SHARE repository. Several new XML-based ADLs such as XML-based Architecture Description Language (XADL) (Zhang, Ding & Li, 2001, pp. 561-566) and Service Oriented Architecture Description Language (SOADL) (Jia, Ying, Zhang, Cao & Xie, 2007, pp. 96-103) may form the basis for this development.

Interfaces can also be represented using UML or other graphical notations. Typical graphical notations of interfaces include the "lollipop" depiction or the expression of an interface as a UML stereotype. Often, these pictorial depictions of interfaces are further defined using a formalized language such as the OMG IDL described earlier (Clements et al., 2002, p. 241). In addition to the visual aid provided by the diagrams, the value of using UML for interface descriptions is that many tools have been developed to read UML and translate the models into XML depictions (XMI) and into executable code. Model-driven Architecture products are available that enable the automatic development of "glue code" between components from the architecture specification (Frankel, 2003).

On the downside, an object-oriented programming development paradigm is assumed. While some generality can by achieved by using packages and subsystems as the main UML building blocks instead of classes and subclasses, some argue that attempting to use UML outside the arena for which it was designed is more trouble than it is worth (Shaw & Clements, 2006, p. 34). This realization, as well as our understanding that whichever description method is chosen must be applied across multiple development cultures, compels us to assert that UML may not be the best way to represent interfaces for SHARE.

Future deployment of the SHARE repository is likely to evolve toward the Service-Oriented Architecture (SOA) of the GIG. SOA has been described as "an ideal vision of a world in which resources are cleanly partitioned and consistently represented" (Erl, 2005, p. 3) and "automation logic is decomposed into smaller distinct units of logic […] known as services" (pp.

23-33). Elements of a service architecture are similar to SHARE concerns—the architecture typically includes a registry of services containing descriptions of services and access information. Mechanisms are provided for service discovery, passing sufficient information about the service back to the caller so that the service can be employed. Advanced concepts include service orchestration for composing higher-order services from component services. The focus, of course, is service reuse, which potentially reduces development and maintenance while improving software reliability and evolution agility.

SOA realization may employ several Web Services standards: Universal Description, Discovery, and Integration (UDDI) for creating service registries; Web Services Description Language (WSDL) for identifying operations offered by services and describing input/output interfaces for those operations; the Simple Object Access Protocol (SOAP) for accessing services and passing data to/from the services; Web Services Business Process Execution Language (WS-BPEL) for describing workflow logic for orchestration of services; OWL for Services (OWL-S), for an ontology of services supporting service advertisement and discovery, description of service operation, service interoperation; Web Services Interoperability (WS-I) profiles for describing collections of Web services specifications at specific version levels; and others. It is interesting to note that the problem of describing Web services in sufficient semantic detail to enable automatic composition of services is similar to the problem of describing software components for reuse.

In Web Service implementations, XML is generally used to hold the information passed across an interface. XML schemas are extensible and easily modified if there is a need to change the standardized format of the data. The above standards for describing and implementing Web Services are XML-based specifications. Additionally, XML is readily digestible by many existing tools and is well enough understood universally to be implemented into new ones. These advantages motivate us to propose XML as the primary notation for documenting metadata, including the interfaces, for the SHARE component specification and ontology project. The flexibility of XML will enable us to incorporate the necessary information to enable capabilities similar to those enabled by ADLs without the high overhead cost of training the end users. Although SOA and Web Services are in a high state of flux as industry standards mature, they present opportunity to create software component specifications in SHARE that can be employed for a number of purposes.

Modeling Software Behavior

In addition to understanding the interfaces for a component, a repository user is interested in the functionality of the software components. In this section, we discuss a number of notations currently used to describe the activities that take place within a component.

In addition to the structural diagramming capabilities provided by the UML, several types of diagrams are used to model dynamic aspects of the system. Methods for formal documentation of behavior provided by UML include sequence diagrams, which may be further amplified using a constraint language such as UML's Object Constraint Language (OCL), collaboration diagrams, and statecharts. Sequence diagrams, or message sequence charts, show the interactions of objects within a component in a time-ordered sequence (Larman, 2005, pp. 222-225). Collaboration, or communication, diagrams also show objects and their interactions but in a more condensed format that tends to lose the visibility of the time-ordered sequencing. State Machine Diagrams, or statecharts, illustrate events and states of objects (Larman, 2005, pp. 485-492). Amplifying information, such as actions triggered by transitions, and activities that take place during particular state conditions is often included.

Each of these UML diagrams sheds light on particular aspects of a component's behavior and could be used to formalize the behavioral descriptions of artifacts incorporated into SHARE. There are a few drawbacks to this approach, however. First, as discussed previously, the use of UML diagrams often assumes an object-oriented development paradigm, which may not be relevant for all SHARE submitters. Second, the UML tools presented are primarily to assist in system development and may not be best suited for asset discovery and retrieval. Repository users are likely to be more interested in a more abstract view of the system than this implementation level information provides. Finally, each of the diagrams only captures a particular "slice," or view, of the software's behavior. For a complete behavioral description, it would be necessary to require each type of diagram plus additional information. This would result in a steep overhead to develop this information for each item contained in SHARE. For these reasons, we do not anticipate incorporating UML activity/state diagrams as the standard representation method for software behavior. However, if these depictions are generated as part of the software engineering development process, they should be included as artifacts in the repository.

In formal specification, system behavior is described using mathematical structures. Formal notations that enable this type of specification include the Vienna Development Method (VDM), Z (pronounced zed), and Alloy. Since the languages are mathematically based, developers can use logic to reason about a formally specified system and sometimes prove its correctness. Application of such techniques generally requires a solid understanding of set theory, logic and other mathematical foundations when learning how to construct specifications in the language. This is one of the complaints about formal languages as well as one of the reasons that the use of formal specification is mostly a topic of research and limited in practical applications to systems or portions of systems with safety-critical reliability demands.

MIT's Alloy project is one of the more successful attempts at making formal methods more user-friendly (Jackson, 2006). Alloy helps users develop the specification by providing a visual simulation of the model. This enables users to recognize when the model is incorrect, and they can then iteratively develop the model in more detail. Alloy also includes an analyzer that automatically checks invariants for inconsistencies in the model. Even with these advances, however, the amount of effort required to specify systems in these formal notations is well above the desired level of effort threshold for the SHARE repository. Therefore, we do not intend to use formal languages to represent software behavior of assets in SHARE.

For SHARE, we do not hope to solve the software composition problem in the near term. Mandating formal descriptions of software behavior for repository items does not seem worthwhile when the composition problem remains unsolved. However, intermediate steps towards formalized behavior descriptions will prove useful in the near term and helpful in advancing towards far-term goals. To this end, we are currently planning to extend the XML-defined metadata to incorporate interface information as well as existing reference architecture information to standardize behavioral descriptions for each artifact entered into the repository. Ongoing advances in service composition in SOAs will also be examined for application to the framework.

Relationships Framework (Ontology)

Rich ontologies capturing the relationships of entities from multiple views have not been applied to software repositories. However, there are many examples of ontology use in the organization of data for different applications. One example is intelligence community synthesis of disparate pieces of information from widespread sources into logical connections in order to

form coherent pieces of knowledge. There are currently several applications designed to collect the data and assist the analyst in drawing relationships among the data. For example, Palantir Technologies has created one such software application to support the DoD intelligence community by providing robust capabilities for managing data from various sources. The Palantir tool is based on user-defined ontologies and supports multiple representation and analysis tools. Graphical representations depict the data items and their relationships with each other, based on the underlying ontology. The analysis tools can be used to form logical links between entities in the database and to detect patterns and irregularities in the data. This rich environment enables multiple search techniques: using keywords, browsing through data tables, and browsing graphical views of the database content based on the relationships of the entities described in the ontology.

For the SHARE research project, these capabilities serve as examples of potential utility of the repository framework by demonstrating the power of formalized semantics. When the framework is in place, technologies such as these can be exploited to gain flexibility in the search options described previously. Similar examples of the use of ontologies to support data analysis exist in other domains, particularly in the medical field. Some background on current and emerging standards for describing rich semantics in data relevant to the SHARE framework is provided in this section.

Semantic Web Techniques

The Semantic Web stack was shown in Figure 3. Several of the components pictured there contribute to stronger semantic description of Web-based resources and are discussed briefly here.

The Resource Description Framework (RDF) is a language for stating assertions in the form of subject-predicate-object triplets. Each of the elements in an RDF statement is an abstract Web resource identified by a URI. RDF and RDF Schema (RDFS) will be investigated for applicability to the SHARE framework to describe taxonomies (class hierarchies) supporting inference and search (Alesso & Smith, 2006). We will also explore the possible benefits of creating RDF expressions for storing SHARE repository data content.

The lower layers of the Semantic Web stack provide the ability to describe information (metadata and schemas) and to express knowledge (assertions). Query languages provide a means to access information. The XML Query language is used to search XML documents by exploiting the hierarchical tree structure of the documents (XPath expressions). The SPARQL Protocol and RDF Query Language provide a means to search RDF expressions by exploiting the subject-predicate-object graph structure of the expressions (pattern matching). If RDF structures prove valuable for describing information in the SHARE repository, the use of SPARQL and other query techniques will be explored.

The Web Ontology Language (OWL) extends RDF/RDFS constructs to provide more precise description of classes, subclasses, and relationships among classes (properties). OWL adds the capability to define local scope of properties, disjointness of classes, Boolean combinations of classes, cardinality restrictions, special characteristics of properties (e.g., functional, transitive, symmetric), and other aspects not expressible with RDF/RDFS (Alesso & Smith, 2006). We will investigate the use of OWL for ontology development for the SHARE framework. Use of OWL will maximize utility by software applications, including use of openly available reasoning engines that can be used to check for ontology consistency and to make inferences about instances in the asset knowledge base.

Rules and rule-based systems provide additional expressiveness in describing the logic of a system. Rules permit software to infer a conclusion from a premise (Alesso & Smith, 2006). Rules may be used in the formalized specification of software assets in the repository to enrich their description, particularly if there is a need to encode business rules, policies, and processes appropriate to the repository (e.g., role-based access). The use of the well-established, Web-based conventions in the information technology community provides a basis for application of a variety of common logical computations. We will be able to employ existing products that can operate on the semantic descriptions using provably correct methods. Cryptologic aspects of the Semantic Web stack cut across all the layers and support such functionality as authentication, encryption, and digital signature (Eastlake & Niles, 2003). We will not address this area directly in the work, but will be creating the semantic basis for implementation of methods such as role-based access and other controls on information content in the repository. Trust is obtained when we can anticipate the actions of a system and have a reasonable expectation that the system will act correctly (i.e., as intended) (Michael, 2008). Trust is often established and maintained through transparency. One of the advantages of the use of the Semantic Web practices is visibility of the information through its description in metadata, semantic descriptions, rules, and computationally sound logic. Clearly, users of the repository will rely on the trustworthiness of the content when obtaining information or artifacts supporting new developments. While we will not address this aspect of the problem directly in the component specification and ontology development, our goal is to make the information as explicit and accessible as possible to humans and machines in order to promote this level of the Semantic Web stack.

Well-defined syntax and semantics for description of metadata, taxonomies, and ontology for the SHARE framework will facilitate development of software applications and user interfaces for working with the repository. By expressing the SHARE component specification and ontology using common Semantic Web elements, the products of our current research will readily support development of various applications including Web Services in an SOA while also providing a basis for future applications employing emerging Semantic Web Services technologies.

Semantic Search

Semantic search methods “augment and improve traditional search results by using not just words, but meaningful concepts” (Alesso & Smith, 2006, p. 201). A prominent approach is Latent Semantic Indexing, which considers documents that share many words in common to be semantically close, without any understanding of the “meaning” of the words. As introduced earlier in this report, other researchers at NPS are developing semantic search capabilities (ReSEARCH) for the SHARE repository that will use the WordNet database to extend this approach to include related words (synonyms, part-of relationships, etc.). For even greater formulation of context, the metadata, taxonomy, and ontology specifications for the SHARE framework discussed above will provide domain-specific semantics that should enable more precise discernment of relevance in the searches. As the formalized semantics of the component specification and ontology are developed, the formalisms will be provided to the ReSEARCH developers to determine if improvements in search precision can be achieved.

Enriched semantic specification of the assets in the SHARE repository will enable users to more readily find resources that meet their need in their context. Extensive work in the Web community is providing tools and techniques that can be applied to the SHARE framework. We will select and apply appropriate techniques to meet the goals of the framework development.

Next Steps

Based on our vision for the framework and the related existing technologies we have summarized, this section lays out our intended path for completing development of the SHARE repository framework.

SHARE Metadata

An initial list of required asset information has been developed by the SHARE Program Office at Naval Surface Warfare Center, Dahlgren, VA. We have developed an XML schema based on this initial list and will complement the metadata fields with the necessary information for filling out the framework. To fill out the data set, we will evaluate known good metadata examples and pull relevant information into the SHARE metadata. We will then ensure that the metadata includes all the information necessary to place the artifact in the appropriate context based on the ontology. In order to promote maximum exposure of SHARE contents, we will also ensure that minimum requirements of the DDMS are satisfied. Based on these considerations, we will develop a practical metadata schema. This will most likely include a core data set and variations for different types of artifacts.

In order to evaluate the completeness of the metadata, we intend to investigate case studies for each phase of the software development cycle. As stated previously, repository users' needs vary greatly depending on their purpose at the time of search. Therefore, we are constructing case studies that capture the potential needs based on users' current development activities. For each of these case studies, the metadata will be evaluated to ensure inclusion of appropriate information for enabling retrieval decisions.

SHARE Software Behavior Representation

For the SHARE software behavior representation, we suspect that the overall goal of implementing formalized representations of standardized software behavior is not feasible in the short term. While we intend to keep the loftier goal in mind, it is likely that an interim step towards standardization of formal software behavior representation will be required.

One near-term solution may be to use available domain information that standardizes descriptions of software functionality. For example, the Common Systems Function List (CSFL), Common Operational Activities List (COAL), and Common Information Element List (CIEL) are leadership-endorsed listings of combat system functionality that can be utilized as an initial characterization of software behavior. We will investigate the use of a subset of these listings in the development of taxonomies for the SHARE repository framework. If we require asset submitters to state the functionality of the components in these terms, we can then build the tools to guide the user in selecting desired behavior in the same terms. We will also explore characterization of software assets based on current and emerging Web Services (e.g., WSDL) and Semantic Web Services (e.g., WS-BPEL, OWL-S) approaches.

SHARE Relationship Framework (Ontology)

The ontology for SHARE will be based on several relationships among the items in the repository, as well as relevant domain architectural descriptions and other information. The types of relationships we are currently exploring are the artifact's place in the software engineering lifecycle, its architectural fit in its original system, its architectural fit in any systems in which it was subsequently used, identification of the component's fit in the Surface Navy Open Architecture reference architecture, and the semantic relationships of various documents in the repository (based on the ReSEARCH work). Each type of relationship will be examined to

determine its appropriate representation form (RDF, OWL, etc.). The goal is to determine representation forms that will best enable tool development, which will in turn support the types of search described in the previous chapters based on the ontology provided.

Future Work

The current research will describe the component specification and ontology desired for the SHARE repository. Further work will be necessary to implement the framework and develop a tool suite that enables the described search capabilities. In the SHARE implementation, additional repository features can be added, such as an Amazon-like "similar results" feature that points people with similar problems to the retrieval of the same files (and other similar recommendations found in Johnson (2007)). In the long term, further work will be required if the intent is to eventually enable automated composition of a system based on reusable components. As mentioned previously, a starting point to accomplish this goal may be to standardize a formal behavior representation of the repository contents.

List of References

Alesso, H.P., & Smith, C.F. (2006). *Thinking on the web: Berners-Lee, Gödel, and Turing.* Hoboken, NJ: John Wiley & Sons.

Belcher, M. (2007). *PEO IWS software hardware asset reuse enterprise (SHARE)*. Information brief.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001, May 17). The semantic web. *Scientific American*, *284*(5), 34-43.

Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., et al. (2003). *Documenting software architectures: Views and beyond*. Boston: Addison-Wesley.

Comprehensive PERL Archive Network (CPAN). (2007). CPAN. Retrieved January 16, 2007, from http://www.cpan.org

Daconta, M.C., Obrst, L.J., & Smith, K.T. (2003). *The semantic web: A guide to the future of XML, web services, and knowledge management*. Indianapolis: Wiley Publishing.

DoD Chief Information Officer. (2003, May 9). *Net-centric data sharing strategy*. Washington, DC: Author.

Deputy Assistant Secretary of Defense. (2007, August 10). *Department of Defense discovery metadata specification (DDMS)* (Ver. 1.4.1). Washington, DC: Deputy Chief Information Officer.

Eastlake, D.E,. III, & Niles, K. (2003). *Secure XML: The new syntax for signatures and encryption*. Boston: Addison-Wesley.

Erl, T. (2005). *Service-oriented architecture: Concepts, technology, and design*. Upper Saddle River, NJ: Pearson Education.

Frankel, D.S. (2003). *Model driven architecture: Applying MDA to ENTERPRISE COMPUTING*. Indianapolis: Wiley Publishing.

Jackson, D. (2006). *Software Abstractions*. Boston: MIT Press.

Jia, X., Ying, S., Zhang, T., Cao, H., & Xie, D. (2007). A new architecture description language for service-oriented architecture. In *Proceedings from the 6th International Conference on Grid and Cooperative Computing* (pp. 96-103). Washington, DC: IEEE Computer Society.

Johnson, J. (2007, October). *SHARE repository component specification: Needs assessment.* Technical Report. Monterey, CA: Naval Postgraduate School.

Larman, C. (2005). *Applying UML and patterns: An introduction to object-oriented analysis and design and iterative development* (3rd ed.). Upper Saddle River, NJ: Prentice Hall.

Martel, C. (2007). *ReSEARCH: A requirements search engine*. Proposal for Future Combat Systems Open Architecture for Naval Sea Command PEO IWS-7.

Medvidovic, N., & Taylor, R.. (2000). A classification and comparison framework for software architecture description languages. *IEEE Transactions on Software Engineering*, *26*(1), 70-93.

Michael, B. (2008, March 19). Perspectives, practices, and the future of building highly dependable and trustworthy systems. Software Engineering Presentation delivered at the Naval Postgraduate School, Monterey, CA.

Object Management Group. (2005). Reusable Asset Specification (Ver. 2.2). Retrieved January 29, 2008, from http://www.omg.org/technology/documents/formal/ras.htm

Sarkar, M., & Brown, H. (1993). Graphical fisheye views. *Communications of the ACM*, *37*, 73-83.

Shaw, M., & Clements, P. (2006). The golden age of software architecture. *IEEE Software*, *23*(2), 31-39.

SourceForge. (2007). *SourceForge.net: Welcome to SourceForge.net.* Retrieved October 8, 2007, from www.sourceforge.net

Szyperski, C. (2002). *Component software: Beyond object-oriented programming* (2nd ed.). New York: Addison-Wesley.

Princeton University. (2006) *WordNet—Princeton University Cognitive Science Laboratory*. Retrieved March 20, 2008, from wordnet.princeton.edu

World Wide Web Consortium (W3C). (1994). *Semantic web stack.* Retrieved February 26, 2008, from www.w3.org/2007/03/layerCake.png

Zhang, B., Ding, K., & Li, J. (2001). An XML-message based architecture description language and architectural mismatch checking. In *Proceedings from the 25th Annual International Computer Software and Applications Conference* (pp. 561-566). Washington, DC: IEEE Computer Society.

Acknowledgements

The work described in this paper is sponsored by PEO IWS. The authors would also like to thank Dr. Mikhail Auguston for his contribution to the preparation of this paper by providing leadership in developing the concept for the repository framework. The opinions expressed in this paper are those of the authors and not necessarily the position of their respective parent organizations or the sponsor of the work described.

2003 - 2008 Sponsored Research Topics

Acquisition Management

- Software Requirements for OA
- Managing Services Supply Chain
- Acquiring Combat Capability via Public-Private Partnerships (PPPs)
- Knowledge Value Added (KVA) + Real Options (RO) Applied to Shipyard Planning Processes
- Portfolio Optimization via KVA + RO
- MOSA Contracting Implications
- Strategy for Defense Acquisition Research
- Spiral Development
- BCA: Contractor vs. Organic Growth

Contract Management

- USAF IT Commodity Council
- Contractors in 21st Century Combat Zone
- Joint Contingency Contracting
- Navy Contract Writing Guide
- Commodity Sourcing Strategies
- Past Performance in Source Selection
- USMC Contingency Contracting
- Transforming DoD Contract Closeout
- Model for Optimizing Contingency Contracting Planning and Execution

Financial Management

- PPPs and Government Financing
- Energy Saving Contracts/DoD Mobile Assets
- Capital Budgeting for DoD
- Financing DoD Budget via PPPs
- ROI of Information Warfare Systems
- Acquisitions via leasing: MPS case
- Special Termination Liability in MDAPs

Human Resources

- Learning Management Systems
- Tuition Assistance
- Retention
- Indefinite Reenlistment
- Individual Augmentation

Logistics Management

- R-TOC Aegis Microwave Power Tubes
- Privatization-NOSL/NAWCI
- Army LOG MOD
- PBL (4)
- Contractors Supporting Military Operations
- RFID (4)
- Strategic Sourcing
- ASDS Product Support Analysis
- Analysis of LAV Depot Maintenance
- Diffusion/Variability on Vendor Performance Evaluation
- Optimizing CIWS Lifecycle Support (LCS)

Program Management

- Building Collaborative Capacity
- Knowledge, Responsibilities and Decision Rights in MDAPs
- KVA Applied to Aegis and SSDS
- Business Process Reengineering (BPR) for LCS Mission Module Acquisition
- Terminating Your Own Program
- Collaborative IT Tools Leveraging Competence

A complete listing and electronic copies of published research are available on our website: www.acquisitionresearch.org

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CALIFORNIA 93943

www.acquisitionresearch.org