NET-CENTRIC ADAPTER FOR LEGACY SYSTEMS

Alan Thomas
Thomas Turner
Scott Soderlund
Naval Surface Warfare Center (NSWC) Dahlgren

ABSTRACT

The Net-Centric Adapter for Legacy Systems (NCALS) is a software technology that makes legacy system data and services available in near real-time to the military Global Information Grid (GIG). The intent of NCALS is to lower the cost and risk, and to decrease the time required for legacy systems to comply with U.S. Department of Defense (DoD) net-centric technical standards. Many different systems could use a common, configurable NCALS software component to comply with these standards. The benefit to the warfighter is improved interoperability with joint and coalition forces.

NCALS enables legacy systems to move to a Service-Oriented Architecture (SOA) compatible with the GIG without requiring a costly and risky re-architecture of their legacy software. In addition, NCALS enables mission critical systems such as weapon systems to segregate their real-time, mission critical software from enterprise integration software. This maintains the safety and security required by such systems, while accommodating rapid changes in Internet-based, enterprise technologies.

This paper will discuss the legacy system challenge and describe a technology prototype developed by the Naval Surface Warfare Center (NSWC) Dahlgren to realize the NCALS concept. The prototype works automatically, behind the scenes, to expose legacy data to the GIG and to make GIG data available to legacy systems.

INTRODUCTION

The beginning of the twenty-first century is an era of surprise and uncertainty, presenting a variety of challenges to the U.S. Department of Defense (DoD). These include: asymmetric operations, non-state enemies, the need to compress mission timelines, and the need to work with a great variety of partners [5]. Meeting these challenges requires great agility. As a result, the DoD Chief Information Officer (CIO) has focused on access to and sharing of information and timely, actionable intelligence among geographically distributed military units, as well as collaborative capabilities [1].

The DoD has developed the concept of Network-Centric (a.k.a. net-centric) Operations (NCO) as a means of meeting these challenges [2-6]. The net-centric approach requires the networking of sensors, decision-makers and weapon systems to enable shared awareness, rapid decision-making, higher operational tempo, increased survivability, and self-synchronization [3]. Self-synchronization occurs when forces are able to coordinate their actions in time with one another. Meeting all these challenges requires timely, complete and accurate information available to all forces.

The premise of net-centric operations is that the "whole of an integrated and networked force is more than the sum of its parts" [2]. This system-of-systems approach demands that we provide warfighters access to timely, relevant and accurate information. Some important attributes required to support NCO are noted in Table 1 [10]. A communications infrastructure, the Global Information Grid (GIG), will network the entire DoD enterprise, serving as a key enabler for net-centric operations [19].

Table 1. Some Key Net-Centric Attributes

Title	Description
Internet Protocol	Network communications
Post in Parallel	Immediate posting of data by Provider
Smart Pull	Data accessible and tagged for discovery
Data Centric	Separate data from applications
Quality of Service	Data timeliness, accuracy, completeness and integrity

In 2003 the DoD published its Net-Centric Data Strategy for managing data in a net-centric environment. The key thrusts of the strategy include [7, 9]: (a) ensuring data are visible and available to the GIG when and where needed for decision-making; (b) annotating all data with metadata

Manuscript received April 29, 2009. This work was supported by Naval Surface Warfare Center Dahlgren under its Discretionary Technical Investment Program.

Alan Thomas, Thomas Turner and Scott Soderlund are with the Naval Surface Warfare Center Dahlgren, Dahlgren, VA 22448 USA. The authors may be contacted at 540-653-7678.

to enable data discovery; (c) publishing of data wherever possible to "shared spaces" on the GIG, ensuring availability to users; and (d) moving from unique "point-to-point" interfaces between individual systems to "many-to-many" exchanges on the GIG. DoD systems must expose their data via data access services to support these thrusts.

To realize net-centric operations, the DoD is working to greatly improve communications capabilities through its GIG initiative, to capture warfighter requirements through Communities of Interest, to provide core "enterprise services", and to identify supporting technical standards through the DoD Information Technical Standards Registry (DISR) [1,9,11,17,19]. These technical standards are aligned with Internet and commercial engineering standards and will support the implementation of Service-Oriented Architectures. Some of the key net-centric standards are shown in Table 2 [11].

Table 2. Some Key Net-Centric Technical Standards

Hypertext Markup Language (HTML)
Hypertext Transfer Protocol (HTTP)
Simple Object Access Protocol (SOAP)
Transmission Control Protocol / Internet Protocol (TCP/IP)
Web Services Description Language (WSDL)
eXtensible Markup Language (XML)
eXtensible Stylesheet Language (XSL)

The authors have observed that one of the most significant obstacles to realization of net-centric operations is the existence of legacy systems within the DoD. Legacy systems are existing DoD systems, which were typically not designed to support net-centric technical standards. For example, the Assistant Secretary of the Navy for Research, Development and Acquisition in 2005 identified 164 legacy systems, 42% of the total number of systems, in the Navy and Marine Corps that will require upgrades to meet the net-centric technical standards. Only 24% of systems were new developments. The remainder of the systems were planned for retirement or were being retained without supporting net-centric standards [8].

The need to substantially change legacy system software results in significant technical, schedule, cost, and programmatic risks for each of these systems [16]. In addition, commercially-available technologies compliant with current net-centric standards (e.g., web services, SOAP and XML) do not support the real-time, deterministic processing required for many applications [20-28]. As a result, it is very risky for mission critical systems (e.g., weapon systems) with real-time and weapon-safety requirements to change their architecture to comply with enterprise standards.

The authors' experience has shown that such a fundamental change on a typical DoD system development schedule invites mission failure. Mission-critical systems require stability in their internal architectures and real-time, deterministic processing for safety, security and certification reasons. On the other hand, net-centric technical standards will have a much higher turnover rate, as they are based on commercial and Internet standards.

The purpose of this paper is to discuss the technical challenges of net-enabling legacy systems and to describe a software technology that enables those systems to implement net-centric standards with reduced risk, cost and schedule.

THE LEGACY SYSTEM CHALLENGE

Legacy systems present a number of technical challenges to achieving net-centric standards compliance. These issues often include: software architectures [16], data formats, external interfaces, and constraints of safety and security. These challenges are discussed below.

Legacy Software Architectures

The typical legacy tactical system in DoD today was not designed to comply with net-centric technical standards in either their internal software architectures or their external interfaces. The authors have observed that many of these systems were not designed to use some of the key Internet-related standards, such as HTML, HTTP, SOAP, WSDL and XML. The reasons for this include: (1) the inability of Internet standards to satisfy real-time, mission critical requirements [20-28]; and (2) the computing standards and commercial products available at the time many of today's military systems were developed [29-31].

The authors have observed two prevalent mechanisms used for communications within legacy military software architectures by server and client software applications: (a) Remote Procedure Calls (RPCs) made using Application Programming Interfaces (APIs), and (b) the Common Object Request Broker Architecture (CORBA) [32]. Both RPC/APIs and CORBA use the Transmission Control Protocol / Internet Protocol (TCP/IP) for message transmission.

Figure 1 depicts the use of RPC over socket-based APIs to support communications between client and server software applications over a network. The figure shows a client application and a server application connected to a common network. The client needs to perform a function on some data, while the server can perform that function.

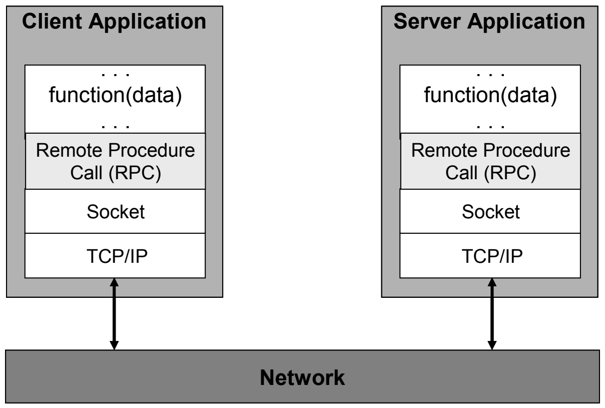

Figure 1. Remote Procedure Call over an Application Programming Interface

To initiate this function call, the client uses an RPC to remotely call the server function. The client uses software "sockets" and TCP/IP to call the function. On the network, TCP/IP is used to transmit the function call and provide it on the proper socket. The server recognizes the RPC, performs the function, and provides the results back to the client.

Figure 2 depicts the use of CORBA to support communications between client and server software applications over a network. CORBA, a specification developed by the Object Management Group [32], is an object-oriented counterpart to a socket-based API. In the object-oriented paradigm, the calling application invokes a "method" on an object. An Object Request Broker (ORB), which is a software implementation of the CORBA specification, keeps track of the locations of software and objects on the network. It determines the location of Object A's implementation and invokes the appropriate method on that object.

Legacy Data Formats and Access

Data in a typical legacy system are held in legacy formats. Often these consist of custom and proprietary data formats and conventions that were selected by the developer to initially implement for each legacy system. Legacy formats may be due to the lack of data standards and/or the lack of interoperability requirements at the time the system was developed. In addition, the legacy data is rarely accessible via standard, net-centric mechanisms (e.g., web services). Accessing and translating legacy data often requires a very large engineering effort [16].

Legacy Point-to-Point Interfaces

Traditionally in DoD, data exchange between two systems has been defined for point-to-point interfaces [7]. These interfaces are numerous and essentially proprietary, because they only work for a pair of systems. On the other hand, net-centric compliance requires a "many-to-many" data exchange approach [7]. This requires a more open approach to defining interfaces, as well as the implementation of net-centric technical standards such as XML.

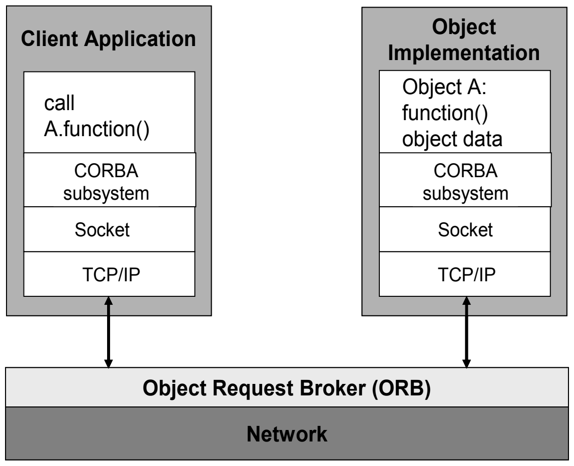

Figure 2. Common Object Request Broker Architecture (CORBA)

Safety and Security Constraints

Legacy military systems typically have important safety and security constraints on their design and performance. For example, weapon systems are concerned with weapon safety requirements, such as maintaining positive control of their weapons under all circumstances. They may also be concerned with issues such as crossing security domains when connecting to the GIG or with other systems with which interfaces are required. As an example, the authors have observed that this is typical for systems on a ship, where a mixture of multiple security domains exists, due to a variety of classification levels, access authorization, and "need to know."

As a result, radically changing the software architectures of such systems greatly increases risk and invites mission critical failure. Despite great advances in software technologies, current enterprise standards, tools and implementations are largely unable to meet the most challenging real-time, deterministic requirements of these critical systems [20-28].

NET-CENTRIC ADAPTER

Net-Centric Adapter Concept

The goal of the Net-Centric Adapter for Legacy Systems (NCALS) is to enable legacy military systems to affordably participate in a net-centric force. Specifically, NCALS minimizes the changes to legacy systems required to participate on the GIG. In addition, it is designed as a common, generic software component that could be used by many different legacy systems. To achieve this, several technical objectives were necessary:

- Enable legacy systems to publish their data and services to the GIG
- Enable legacy systems to subscribe to GIG data and services
- Reduce the software development effort required to implement net-enable legacy systems
- Architect the NCALS software as a common, configurable component

The NCALS software concept is illustrated in Figure 3. It serves as an automated, two-way gateway between a legacy system and the GIG. As such, it works behind the scenes in an automated fashion to expose data from legacy systems to users of the GIG. In addition, it must be configurable, as a common software component, to support a variety of legacy systems needs, as well as portable across a variety of computing platforms. Lastly, its architecture must be scalable to accommodate the net-centric data requirements of many different legacy systems.

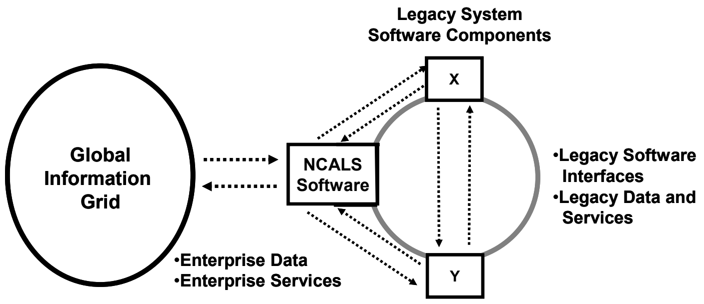
Legacy System Software Components
X
Global Information Grid
NCALS Software
•Legacy Software Interfaces
•Legacy Data and Services
•Enterprise Data
•Enterprise Services
Y

Figure 3. NCALS Software Concept

A legacy system consists of legacy software components running on hardware, typically on a Local Area Network (LAN). These software components communicate with one another via legacy software interfaces (e.g., APIs and/or CORBA) and hold data in legacy formats.

The NCALS software uses the existing legacy software interfaces to obtain legacy data, transform it into net-centric standard formats and publish it to the GIG in compliance with net-centric standards. Likewise, it transforms GIG data into legacy data formats and injects it into the legacy system via its existing software interfaces.

Regardless of the particular domain, data and services of a legacy system, much of the NCALS functionality is common. However, it is configurable to accommodate the specifics of a particular legacy system operating in its specific domain of operations.

Since NCALS is designed to enable net-centric operations, it focuses on enabling legacy systems to interoperate with the GIG. As a result, it provides a service-oriented architecture connection for a legacy system to the rest of the DoD enterprise on the GIG. However, it does not modify the legacy components to comply internally with the net-centric technical standards. It allows the legacy system architecture to remain largely undisturbed.

Figure 3 assumes that the GIG provides a web server as a "shared space" on the GIG. However, depending on the legacy system and military platform (e.g., a ship) requirements, it may be necessary to couple NCALS with a web server as a package for the system or platform. This is depicted in Figure 4. The arrows in this figure denote functional interfaces that utilize a LAN for connectivity.

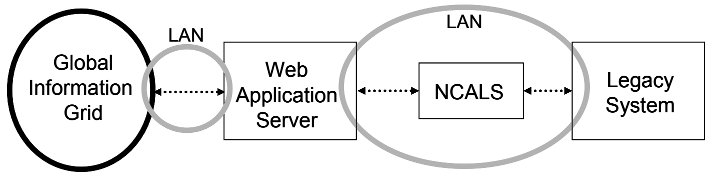

Figure 4. NCALS Coupled with a Web Server

Security Considerations

Since we are considering the requirements of military systems, it is necessary to address security. Integrating NCALS with a legacy system does not require a major change in legacy system architecture for security. Existing secure guards that ensure secure data flow across a security domain can be used as shown in Figure 5.

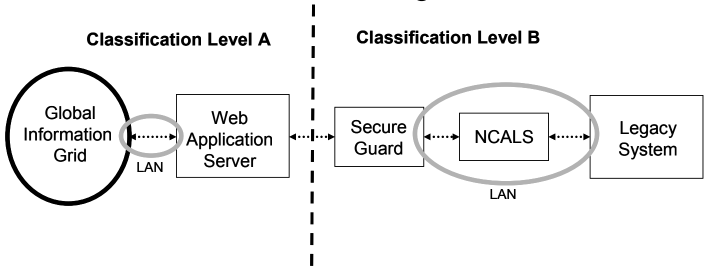

Figure 5. NCALS with a Secure Guard

However, connecting some legacy systems to the GIG may require a secure guard where one was not previously required. Note that the new connection brings the security requirement, no matter how the legacy system complies with net-centric technical standards. It is not a result of using NCALS to comply with those standards. Regardless, some legacy systems will certainly need a secure guard to connect to the GIG. Future plans to address this need are briefly described later in this paper.

Software Prototype

NSWC Dahlgren developed a software prototype of NCALS in a three-phased prototyping effort. Phase I demonstrated an initial prototype that performed legacy/enterprise data transformations in near-real-time with simulations of legacy systems. Phase II matured the prototype architecture and functionality, and demonstrated it with an existing Navy tactical system. During Phase III, we have demonstrated NCALS coupled with a standard web server and created a portable demonstration capability.

The NCALS prototype is implemented in the Java programming language, to promote portability and ease of implementing the Internet-based net-centric standards. It can be configured for data transformations between legacy and net-centric data formats using eXtensible Stylesheet Language Transformations (XSLT) or using custom transforms based on software object classes. In addition to APIs and CORBA (multiple versions), it can support software interfaces that use the Java Messaging Service (JMS) or shared files. Interfaces can also be rapidly customized to other types. Legacy data formats include API parameters (fields) and objects referenced in CORBA interface descriptions.

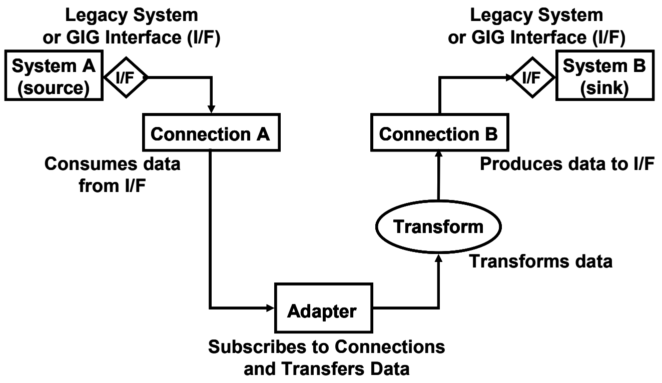

Figure 6. Generic Software Adapter Design

NCALS utilizes a generic software "adapter" design to establish connections between legacy systems and a GIG or platform web server and to make the appropriate data transformations. The unidirectional adapter design is shown in Figure 6. This design enables either a legacy system or a GIG interface to serve as a data "consumer" or a data "producer." The adapter handles connections to both the consumer and the producer. It subscribes consumers of data to appropriate connections, transforms legacy data formats obtained from legacy software interfaces, and transfers data between the two connections.

Two-way interfaces between two systems are implemented by instantiating two, one-way adapters, as shown in Figure 7. So, both a legacy system and the GIG can serve as both a producer and consumer of different types of data, as appropriate.

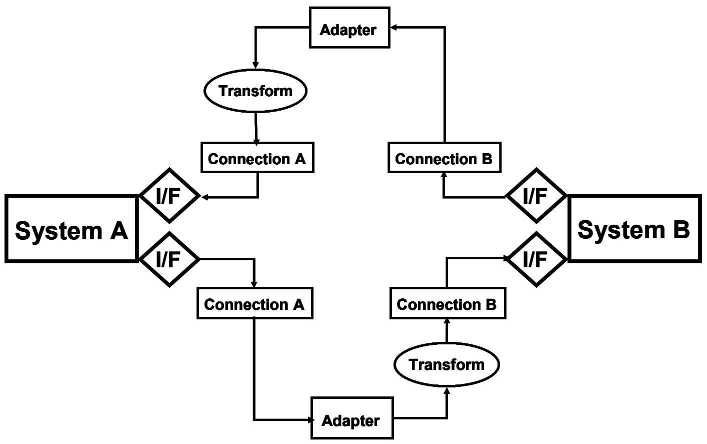

Figure 7. Instantiation of Two, One-Way Adapters

Relationship to Prior Work

Prior to developing an NCALS prototype, NSWCDD conducted an evaluation of Commercial Off-The-Shelf (COTS) tools that claimed the potential to solve the technical challenges described above. The evaluation started with a market survey and ended with a detailed, hands-on evaluation of the most promising tools. This evaluation concluded that most enterprise integration software applications and frameworks are focused on rapid, optimized integration with back-end, legacy databases. These tools do not provide an easy and convenient way to integrate with legacy software-to-software interfaces. While these tools tend to implement good XML-to-XML data translation capabilities, they provide much less capability for legacy data formats [15].

Using these tools with non-XML, proprietary, legacy data formats requires a very significant amount of custom programming [15]. In addition, we found the most capable tools in this area to be expensive [15].

NCALS, on the other hand, was developed with a specific focus on addressing the problem of enabling legacy military systems to comply with net-centric standards. As such, it specifically addresses the common types of legacy software-to-software interfaces and translation of legacy data formats to XML-based formats.

NCALS provides a flexible configuration capability for legacy software interfaces and data formats at a very low level (e.g., the socket connection level and the byte-level for data). Its design in this area is much more generic and easier to use than the COTS tools.

NCALS has applied a number of software design patterns to a much larger scope than seen previously: the level of an entire legacy system or enterprise system, as well as to that of major software components. Traditional design patterns are applied to small software components at the level of classes and methods [13, 14].

NCALS can be considered an application of the Adapter Pattern [13] at this higher level. In addition, NCALS applies the message aggregator, channel adapter, message broker, and publish-subscribe channel patterns [14] at the system level. While NCALS leverages middleware standards (e.g., CORBA), it operates at a level above traditional middleware [29, 31].

In applying these design patterns at the enterprise system level, NCALS provides adaptation of services and data for a system composed of multiple, heterogeneous software components employing a variety of standard and non-standard communication mechanisms. For example, some legacy military systems began using commercial Object Request Brokers (ORBs) when the CORBA specification was mature but still evolving. As a result, they implemented proprietary object directories. When CORBA evolved to standardize naming services [32], the legacy systems were too far along in development to implement the standardized services.

NCALS also supports legacy implementations that are no longer supported by commercial enterprise standards. For example, the Java programming language version 1.5 implemented support of a new version of CORBA in a way that was not backward compatible with older CORBA implementations [32-34]. NCALS supports both the newer and the older legacy implementations of CORBA.

NCALS Demonstration

In 2007, NSWCDD demonstrated the NCALS prototype in the domain of a legacy Strike Warfare system engaging a time critical land target as tasked by a joint service Command and Control (C2) system. The configuration for this demonstration is depicted in Figure 8. The arrows in this figure represent functional interfaces between systems or components.

In this demonstration, the NCALS prototype tapped into API and CORBA software interfaces to obtain legacy data, automatically transformed this data to a net-centric XML-based data format, and transferred this data to a simulated C2 system on a simulated GIG network. Likewise, enterprise data from the C2 system was transformed and transferred to the legacy system via its internal software interfaces.

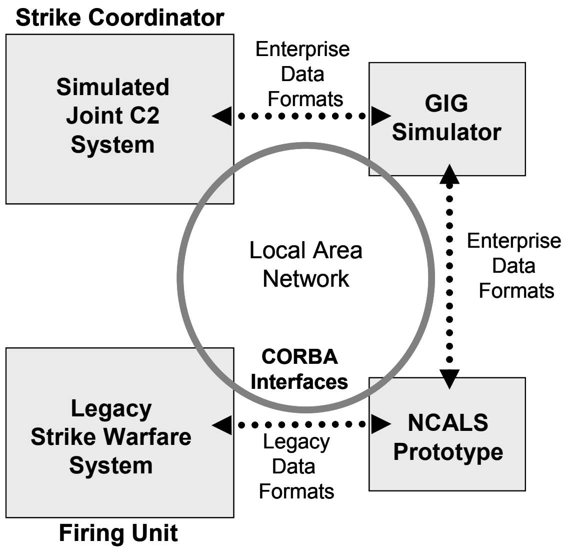

Figure 8. 2007 NCALS Prototype Demonstration

One very significant observation is that no legacy software source code was modified for this demonstration. The only change required to the legacy system was a modification of its secure router's configuration to allow NCALS to connect to its internal network. In 2008, NSWCDD enhanced the NCALS prototype to support web services via a standard web server, as used on the Internet. A simulation-based, portable demonstration was also developed.

FUTURE DIRECTIONS

Near-term plans for NCALS include preparing the current prototype for use by DoD acquisition programs in their legacy systems. Now that many critical functions have been prototyped and demonstrated, NSWC is in the process of collecting baseline NCALS software performance information, including message throughput and latency.

A configuration tool for NCALS is currently being developed. This tool will simplify and automate the configuration of NCALS to meet the needs of a large variety of legacy systems. In addition, the prototype will be further matured in 2010 and packaged as a beta version for legacy system developers to try out.

Beyond these near-term plans, the authors desire to integrate NCALS with a cross-domain security product. This integration will enable legacy system developers to connect higher-security systems to the GIG. One candidate to serve as the secure guard is the Inter-LAN Socket Connection Manager (ILSCM) developed at NSWC Dahlgren. The ILSCM is a Government Off-The-Shelf (GOTS) software component that performs bi-directional XML message validation and medium assurance guard functions. ILSCM, like NCALS, is not tailored to the needs of any particular program. It is versatile and fully scaleable to support the needs of any system seeking an affordable bi-directional guard for XML-based messaging. [12]

The authors would like to explore several additional applications of the adapter concept. One possible extension is that of an aggregator and broker at a domain or Community of Interest (COI) level. This would support having sets of multiple legacy systems that operate in a particular domain or COI integrated with the rest of the GIG via a "domain adapter/broker."

Other potential applications include: using NCALS to integrate legacy software components with new or more modern software components; near real-time data format translation; and leveraging Semantic Web technologies to enable service discovery on the GIG.

CONCLUSION

This paper has discussed the technical challenges to net-enabling legacy military systems and has described NCALS as a potential solution. When configured properly for a particular system and its data requirements, NCALS works automatically, behind the scenes, to expose legacy data to the GIG. It can also inject GIG data into a legacy system via existing software interfaces in existing data formats. This has great potential in DoD to reduce the cost, schedule and technical risks of legacy system compliance with the net-centric technical standards.

REFERENCES

[1] DoD Chief Information Officer (CIO), *Enabling Net-Centric Operations*, downloaded on June 14, 2008 from http://www.defenselink.mil/cio-nii/

[2] Office of the Secretary of Defense (OSD), *The National Defense Strategy of the United States of America*, March 2005

[3] Joint Staff, *An Evolving Joint Perspective: U.S. Joint Warfare and Crisis Resolution in the 21st Century*, 28 January 2003

[4] The White House, *National Security Strategy of the United States of America*, March 2006

[5] OSD, *Quadrennial Defense Review (QDR) Report*, February 6, 2006

[6] Chairman of the Joint Chiefs of Staff, *National Military Strategy of the United States of America: A Strategy for Today; A Vision for Tomorrow*, 2004

[7] DoD CIO, *DoD Net-Centric Data Strategy*, May 9, 2003

[8] SPAWAR Systems Center Charleston, *FORCEnet Compliance Category Report*, March 21, 2005

[9] A. J. Simon, "Overview of the Department of Defense Net-Centric Data Strategy", in *Crosstalk: The Journal of Defense Software Engineering*, July 2006

[10] OASD(NII)/CIO, *Net Centric Attributes List*, Feb. 17, 2004

[11] DoD IT Standards Registry (DISR) Online, https://disronline.disa.mil/a/DISR/index.jsp, accessed on June 16, 2008

[12] J. Gray and J. Valdivia, "Inter-LAN Socket Connection Manager", Poster, *IEEE International Conference on Technologies for Homeland Security*, May 12-13, 2008

[13] A. Shalloway and J.R. Trott, *Design Patterns Explained: A New Perspective on Object-Oriented Design*, Addison-Wesley, 2002

[14] G. Hohpe and B. Woolf, *Enterprise Integration Patterns*, Addison-Wesley, 2004

[15] D. Cozart-Amos, S. Soderlund, A. Thomas and T. Turner, *Commercial Off-The-Shelf (COTS) Product Evaluation Report for Net-Centric Adapter for Legacy Systems (NCALS)*, Internal NSWCDD Report, 13 February 2007

[16] G. Lewis, E. Morris and D. Smith, "Analyzing the Reuse Potential of Migrating Legacy Components to a Service-Oriented Architecture," *Proceedings of the Conference on Software Maintenance and Reengineering 2006 (CSMR`06)*, IEEE Computer Society, 22-24 March 2006

[17] F. Criste, Director, Communications Programs, OASD(NII)-DASD(C3, Space & IT Programs), Presentation: "Implementing the Global Information Grid (GIG): A Foundation for 2010 Net Centric Warfare (NCW), 18 May 2004

[18] U.S. Department of Defense, "Global Information Grid" (Section 7.2), *Defense Acquisition Guidebook*, 16 December 2004

[19] U.S. Department of Defense (DoD) Chief Information Officer (CIO), *Department of Defense Global Information Grid Architectural Vision: Vision for a Net-Centric, Service-Oriented DoD Enterprise*, Version 1.0, June 2007

[20] L. Whitt, "Why is C4I Software Hard to Develop?," *12th International Command and Control Research and Technology Symposium*, June 2007

[21] K. Lund, A. Eggen, D. Hadzic, T. Harfsoe and F.T. Johnsen (Norwegian Defence Research Establishment), "Using Web Services to Realize Service Oriented Architecture in Military Communications Networks," *IEEE Communications*, October 2007

[22] C. Kohlhof and R. Steele, “Evaluating SOAP for High-Performance Business Applications: Real-time Trading Systems,” *Proceedings of the 2003 International WWW Conference*

[23] R. van Engelen, “Pushing the SOAP Envelope with Web Services for Scientific Computing,” *Proceedings of the International Conference on Web Services*, 2003

[24] R.A. van Engelen, “Constructing Finite State Automata for High Performance Web Services,” *IEEE International Conference on Web Services*, 2004

[25] Q. Duan, Service-Oriented network Description and Discovery for High-Performance Grid Computing,” *IEEE International Conference on Service-Oriented Computing and Applications*, 2007

[26] K. Benedyczak, A. Nowinski, K. Nowinski and P. Bala, “UniGrids Streaming Framework: Enabling Streaming for the New Generation Grids,” *Applied Parallel Computing. State of the Art in Scientific Computing*, Springer Berlin / Heidelberg, 2008

[27] C. Demarey, G. Harbonnier, R. Rouvoy and P. Merle, “Benchmarking the Round-trip Latency of Various Java-Based Middleware Platforms,” *Studia Informatica Universalis regular Issue*, Vol. 4, No. 1, May 2005

[28] M.B. Juric, I. Rozman, B. Brumen, M. Colnaric and M. Hericko, “Comparison of Performance of Web Services, WS-Security, RMI and RMI-SSL,” *The Journal of Systems and Software*, Vol. 79, 2006

[29] Bakken, D., “Middleware,” Chapter in *Encyclopedia of Distributed Computing*, J. Urban and P. Dasgupta, eds., Kluwer Academic Publishers, 2001

[30] P. Verissimo and L. Rodrigues, *Distributed Systems for System Architects*, Kluwer Academic Press, 2001

[31] Emmerich, W., Aoyama, M., and Sventek, J., “The Impact of Research on Middleware Technology”, *SIGSOFT Software Engineering Notes* 32, 1, Jan. 2007

[32] Object Management Group, *The Common Object Request Broker: Architecture and Specification*, Revision 2.3.1, October 1999

[33] Sun Microsystems, “Changes in CORBA Features between J2SE 1.4.x and 5.0,” *Java 2 Platform Standard Edition 5.0 Development Kit*, downloaded from http://java.sun.com/j2se/1.5.0/docs/index.html on April 10, 2009

[34] Sun Microsystems, “Official Specifications for CORBA Support in J2SE5.0,” *Java 2 Platform Standard Edition 5.0 Development Kit (JDK 5.0)*, downloaded on April 10, 2009 from http://java.sun.com/j2se/1.5.0/docs/guide/idl/compliance.html