March 28, 2005

UNCLASSIFIED

Perpetual Enterprise Management Service (PEMS) for Next Generation SOA-based Command & Control Systems

Topic Area:	Experimentation or Assessment, Tools, and Metrics
Author / POC:	Erik King
Organization:	Science Applications International Corporation (SAIC)
	C4I Advanced Systems Division
Address:	Mail Stop E-2-5
	8301 Greensboro Drive,
	McLean, VA 20194
Telephone/Fax:	703.676.6513 / 703.676.5670
Email:	kinge@saic.com

An Employee-Owned Company

This page intentionally left blank.

ABSTRACT

United States Department of Defense (DoD) Services and Agencies are building and deploying Next Generation Command & Control (NGC2) systems that are based on highly distributed Service Oriented Architecture (SOA) applications. These new systems do not have 'end-to-end' monitoring and reporting capabilities to establish realistic Service Level Objectives (SLO) and Agreements (SLA) that are needed to support this concept. As SOA and Web technologies proliferate, a service is required that collects and correlates business logic, platform, and network component metrics for problem analysis, resolution, and SLA establishment. Traditionally Combatant Commands have relied on disparate software, system, and network engineers to pinpoint and resolve operational problems in fielded Command & Control (C2) systems. As these traditional client/server systems are upgraded to SOA-based applications, the highly distributed nature of this new Net-Centric Warfare (NCW) capability will require the three functions to merge into one. A Perpetual Enterprise Management Service (PEMS) is needed that enables the new function to effectively manage the emerging complexity that SOA-based C2 software will bring under new programs including Joint Command and Control (JC2) and Net-Centric Enterprise Services (NCES).

This page intentionally left blank.

Table of Contents

List of Figures

This page intentionally left blank.

1. INTRODUCTION

1.1 Proliferation and Popularity of SOAs

One measure of success in the information technology field is the reflection of how a single term nearly non-existent several years prior, is nearly ubiquitous today. Web Services and SOA are two such terms. Visit any local bookstore and you will see the Computer section dominated with books that espouse how to implement various technologies that conform to this new Information Technology (IT) model. You can look to the commercial world to further support this measure. Companies such as *Google*, *Ebay*, and *Amazon* are transforming their 'newly formed' organizations to exploit the power of SOA.

eBay, the world's premier Internet auction house, has invested heavily in Web Services, and has recently begun to reap the benefits of this approach. Forty percent of its current listing business (representing over 1 billion transactions per month) is conducted via Web Services[1]. For a company earning $8 billion in revenues a quarter, that is a significant investment in a technology that did not exist five years ago. It is easy to understand why *eBay* has invested so heavily in building a state-of-the art Web Services platform not only for its customers, but third party vendors as well. The ability to shorten the time of creating, managing, and fulfilling *eBay* listings through machine-to-machine transactions has a direct impact on the bottom line. *eBay* has successfully implemented Web Services in a way that has heightened the transaction 'fervor' within its marketplace. Traditional brick-and-mortar stores who turned to the Internet only a few years ago to sell their products, can use *eBay* tools to liquidate product at the optimal time without going through the hassle of using simple Web page interfaces.

As a result of this commercial popularity, it is no wonder that the DoD C2 community has taken a hard look at how similar technology models can be harvested to solve more complex problems.

1.2 DoD C2 View on SOA Today

A key feature of a SOA is the ability to decentralize and proliferate technology components in a manner that is more efficient with the structure of the business organization itself. Organizational 'functions' can be represented through a series of Web Services or SOA components that can later be synchronized into a larger application. The application might be visualized through a Web portal, or some refined user-centric capability. The flexibility makes the SOA model attractive to DoD C2 applications, particularly those of a 'strategic' nature, because the owners of these functions, and associative data, can focus on building interfaces, and not the overall infrastructure that binds it.

The DoD has geared its C2 Transformation policy and key technology initiatives around this theory. The Global Information Grid (GIG), and its NCES, represent the network and corresponding infrastructure through which future C2 components will conduct business. Individual Services and Agencies within the Department will focus on building components and interfaces that plug into this infrastructure. NCES will provide methods for disparate entities to find, bind, and execute many different functions that collectively will comprise the C2 system of tomorrow.

1.3 Growing field of Web Service Orchestration

One advancement in the Web Services field that has recently empowered the movement to SOA within the DoD C2 community is the promise of Web Service Orchestration. A number of commercial standards have emerged that extend the reach of simple Web Services Definition Language (WSDL) to that of a true business process workflow activity. One standard is called Business Processing Execution Language for Web Services (BPEL4WS). As Figure 1 illustrates, BPEL4WS allows for control logic to be established and called via eXtensible Markup Language (XML) from different Web Services to perform a given set of activities[2].

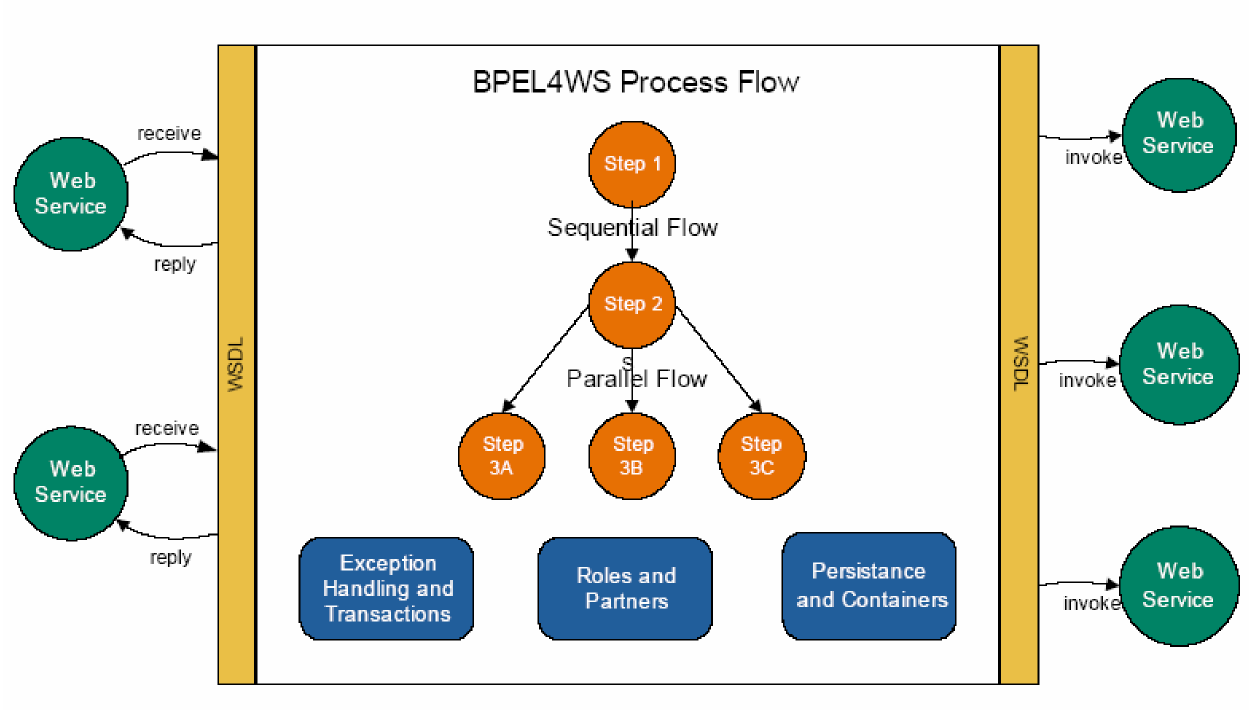


Figure 1: Process Flow via BPEL4WS

The process flow, combined with other standards such as WS-Coordination, WS-Transaction, and the tools invoked to instrument this activity, is typically referred to as Web Services Orchestration. This is a key step forward in the SOA model, where Web Services are not only advertised and self-described, but available for dynamic interconnection with other Web Services, giving rise to the notion that applications can be dynamically generated and potentially implemented with a shortened integration schedule.

However, all of these extensions and refinements within the Web Services industry begs the question: "*Who is managing all of this?*". More importantly, is the pace of traditional enterprise management software and techniques keeping pace with this advancement in the field of dynamic component-built software? The focus of this paper is on this very subject: how to manage the proliferation of Web Services and SOA technology that is taking place within the DoD C2 arena as a result of DoD transformation initiatives such as NCES, the GIG, and future JC2 Mission Capability Packages (MCPs) that are likely to employ these techniques.

2. CURRENT APPROACH TO ENTERPRISE SYSTEMS/SERVICE MANAGEMENT

2.1 Classic 3-tiered Model

Currently there are three primary focus areas, or ‘tiers’, of software that are used to manage the end-to-end user experience that is involved with any SOA application. They revolve around the tiers that support every Web-based user experience, namely the underlying network, the computing platform, and the applications or components that are stitched together to comprise a business logic thread. Figure 2 depicts some of the more popular commercial vendors, and how they typically support these areas.

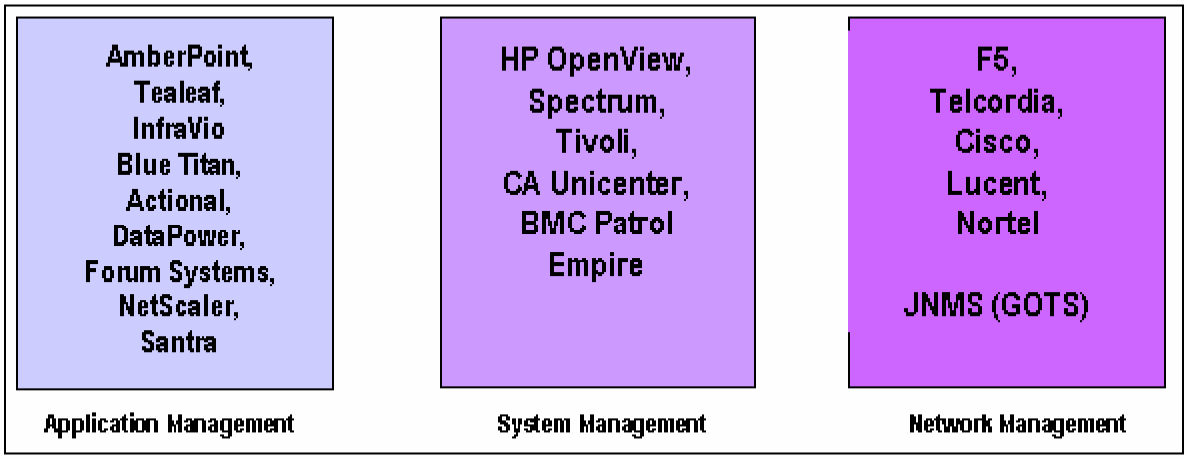


Figure 2: Classic Grouping of Enterprise Management Software

It is important to note that many of these firms are in the process of integrating different aspects of their software, making them more ‘open’ by providing Web service hooks, and developing capability that gives an administrator of their technology some insight into the other areas. However, there is little in the way of technology that truly binds these three without incurring a proprietary model. This is an emerging area that is some way off from becoming a commodity.

2.2 Current Roles with the DoD C2 Community

Due to the popularity of client/server architectures over the last ten years, the classic tiers of enterprise management software have molded a set of key roles that line up directly to the three areas.

There is typically a C2 *Network Administrator* (Local Area Network [LAN] or Wide Area Network [WAN]) that uses a set of network management tools that may be bundled up into a Government off-the-shelf (GOTS)-based system such as the Joint Network Management System (JNMS) to examine the transport medium, its traffic, and corresponding payload. Little regard is placed on the actual application at this level. The primary role is to make sure the network pipe is up, and that the traffic is running smoothly across it.

The *System Administrator* is concerned primarily with the computing devices and platforms that host the business applications involved in a DoD C2 system. This might extend to the operating system software and critical low-level processes running on the devices within a given DoD enclave. There is a vast array of commercial software that is used by the system administrator to manage these devices, with emphasis on key items such as Central Processing Unit (CPU) and Memory utilization, hard disk availability, and load balancing metrics among others. Again, little focus is placed on what the end-user is experiencing at this level.

The third key role is that of the *Software Engineer*. Application design, development, and implementation are the typical activities involved in this role. The end-user experience is of utmost concern, and the components built are designed to support the end-user from a functional perspective. What is different in the evolving SOA model is that more and more this role is being reduced to the development of key functions or capability that is then exposed on the network in the form of Web Services. The application development is being refined, using techniques such as Web Services orchestration, to the interfacing of components built by a variety of individuals. Where traditionally (under a client/server model) the software engineer would play a more active role in the platform and network underpinnings of his/her application, that function has essentially been outsourced in a SOA.

These three roles, which have evolved through specialization over the last ten years, are now being challenged as the decentralization of applications, and the somewhat boundless nature of a SOA-based user experience makes it difficult to pin down how best to coordinate among these administrative functions.

3. C2 PROBLEM DOMAIN

3.1 Management Issues with a Highly Distributed SOA Model

The ‘fog of war’ that has driven the need for a complex and distributed computing model to support the sophisticated tools that are needed for C2, have given rise to a ‘fog of tools’, where vendor claims are distorted through the lens of any given observer. As SOA technologies proliferate, the lexicon is itself shifting and being continuously defined. Attempts to manage what constitutes the full user-experience fall typically into the marginal attempts to piece together the three traditional views or by implementing user-based tools (Such as *Mercury Interactive’s LoadRunner*) that focus on transaction-based testing by emulating a series of user threads. Unfortunately, these transaction-based tests, while providing a sense of whether things are performing good or bad at any given moment, do not provide the in-depth transparency that the traditional tools inherently provide in their respective classes.

Management issues include the following:

- What components comprise my user’s experience?
- How are they inter-connected?
- Where does my business logic physically and logically exist at any given moment?
- Where is the weakest link in the chain of components?
- Can I truly extract myself from the network and platform concerns and let someone else manage those, and focus on Web Services Management (WSM) approaches alone?

All of these are vexing questions and point to the need for a solution that integrates the best of the three tradition administrative technology and roles.

3.2 Need for an Integrated Solution

To quickly assess and resolve system issues, a capability is needed that correlates the logic, hardware, and network dependencies into a single view that corresponds to mission threads being executed in a C2 SOA-based application. Figure 3 illustrates the complexity of the problem. Here the three key layers are shown on the right. What is implied in this diagram is that we have a set of tools that is collecting information using the best-of-breed commercial software that exists in the three spaces. We see in the layers that a group of Web Services is being connected to form a given user experience, and that the applications reside on a set of computing devices that are further hosted on a series of networks. As we move to the left, we can see that our commercial software has instrumented these components, and that we can have a variety of service levels being indicated through the stoplight metaphor being depicted. What is needed is a solution that combines these characteristics into a single view, correlating them through a PEMS such that the upstream user experience can be extrapolated. This approach would provide us with a fundamentally more rich and transparent method of both determining what and where a problem is occurring, and its impact on what the user perceives. In this example, we show how several components at different layers may be operating within a service level agreement, but the overall user experience has failed.

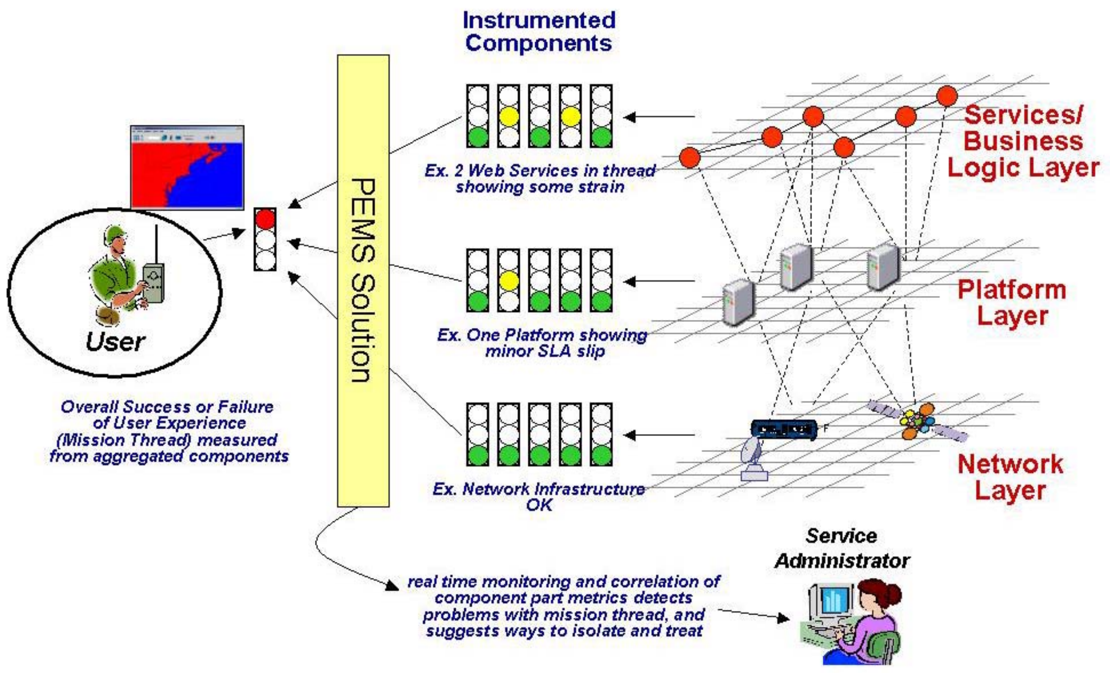

Instrumented Components
Ex. 2 Web Services in thread showing some strain
Ex. One Platform showing minor SLA slip
Ex. Network Infrastructure OK
PEMS Solution
User
Overall Success or Failure of User Experience (Mission Thread) measured from aggregated components
Services/ Business Logic Layer
Platform Layer
Network Layer
Service Administrator
real time monitoring and correlation of component part metrics detects problems with mission thread, and suggests ways to isolate and treat

Figure 3: Where the Integrated Solution is Needed

The PEMS capability (or service) needs to allow the user to define and model a mission thread or user experience that consists of multiple Web service, platform, and network components; then provide capability to dynamically instrument, collect, and aggregate metrics from these parts to predict and address potential failures in the threads in a real-time environment. The real time monitoring tools and associated Web Services should allow users and developers of SOA services (e.g., NCES, ForceNet, and Army Future Combat Systems [FCS] consumers) to determine the underlying reliability and usability of the net-centric infrastructure building blocks where upstream C2 user applications are based.

3.3 Evolution of the 'Service Administrator'

A role quickly emerges from the workflow presented in Figure 3 that combines the understanding and interconnection of network, platform, and business logic elements. A Service Administrator is an individual who will either coordinate with embedded network, systems, and software engineers who are working together to manage the net-centric infrastructure, or become the end-to-end manager of equivalent BPEL4WS constructs and key Web Services that comprise the bulk of a C2 user's experience. By providing a tool that contains the PEMS capability, this individual would represent the equivalent of a technology combat support component to front line consumers of SOA-based NCW applications.

4. THE PERPETUAL ENTERPRISE MANAGEMENT SERVICE (PEMS) CONCEPT

4.1 Thread Management Principle

One thing that is commonplace in an application constructed through an SOA implementation is the fact that many services within it are reused as a user navigates through the various functions to complete a task. Due to the nature of component-built applications development, one can typically derive a handful of key threads that comprise the bulk of any given user experience. A *Thread Management Principle* might therefore be defined as the following

> *For any given application, a limited number of key business logic threads can be identified that comprise approximately 90% of the entire user experience. If the threads can be represented via a logical and subsequent physical model, then perpetually monitored via an integrated tool, fundamental distributed enterprise management of the user experience can be satisfied.*

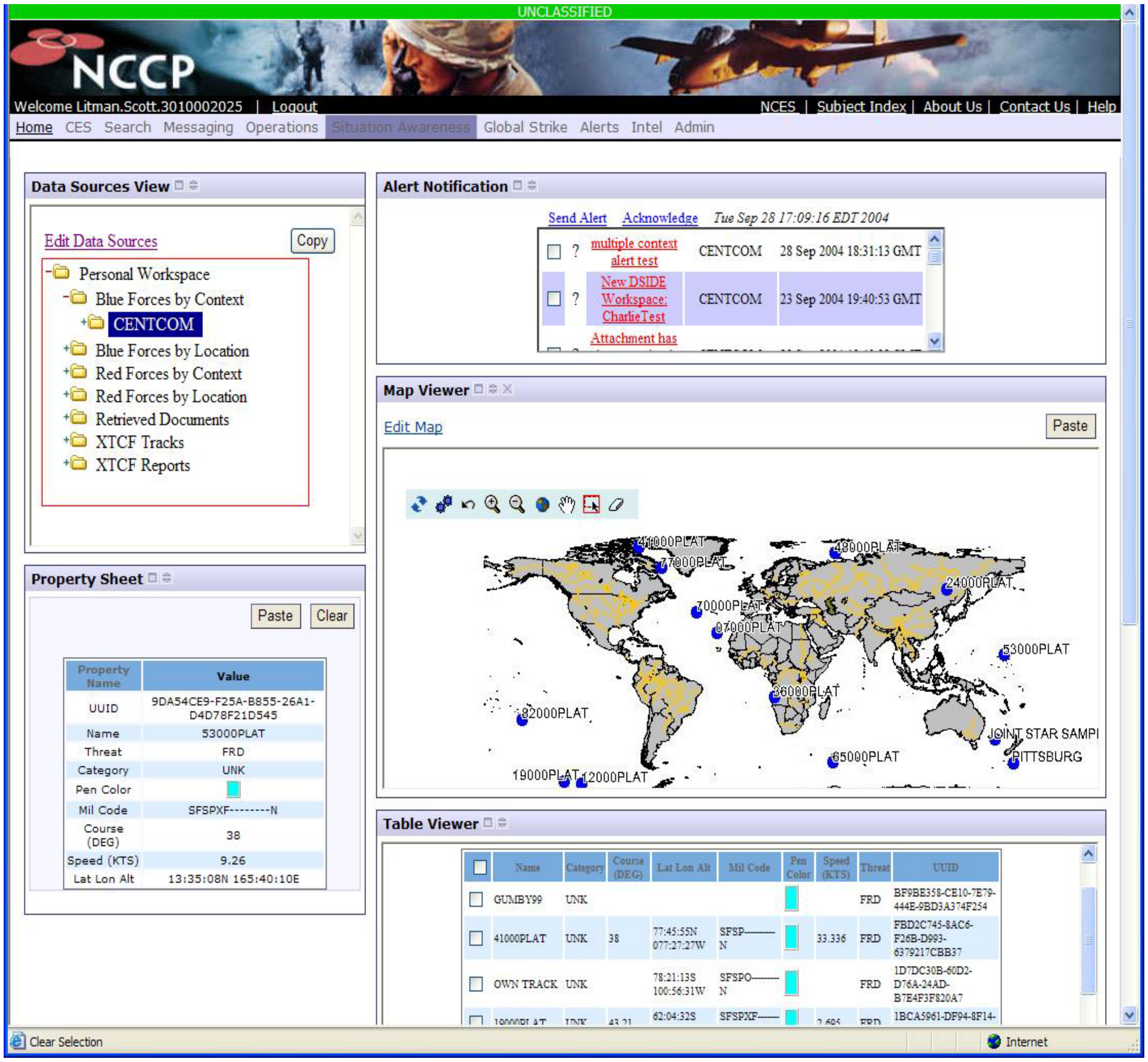
Let us consider eBay again as a model. One of the reason's eBay's Web commerce platform is so successful is the fact that all of their available Web Services (hundreds of them) can be boiled down to the following 6 primary entities[3]:

- **Item**—something sellable on eBay.
- **Listing**—Noun—an entry on eBay with one or more items; Verb—the action of creating such a listing. An auction is a type of listing that enables competitive bidding.
- **Categories**—a hierarchical set of groups on eBay in which items of a similar nature are listed.
- **User**—someone who has registered with eBay. There are user roles such as bidder, buyer, seller, storeowner and application developer.
- **Transaction**—the data for the purchase of one or more items by one buyer from one listing. Some listings enable a seller to offer multiple items in the same listing; thus there could be multiple buyers purchasing items from the same listing and therefore multiple transactions for the same listing.
- **Feedback**—an eBay mechanism or system by which one user may rate another user, enabling other users to know how well or how poorly a transaction went.

From these entities, any business process can be derived or constructed. This is a typical approach to many successful information system implementations. There is a core simplicity that underlies the complex nature of what is being performed behind the scenes. The ability to articulate, normalize, and streamline a complex set of functions in an easy-to-understand model is what attracts subsequent builders to the technology in the first place. In the case of Web Services, this is even more essential, since the orchestration of components can only be done if the functions themselves are of a relative atomic nature, stripped away of extra programmatic trappings, exposing a simple, direct logical construct that has a clearly understandable pedigree.

4.2 Deriving the Logical Model

As part of a research and development effort, from which this paper is based, a key thread from the *Defense Information Systems Agency (DISA)* 2004 *Net-Centric Capabilities Pilot (NCCP)*, (aka *Oktoberfest*), was modeled as input into the PEMS solution. Figure 4 illustrates the end-user portal application, from which a number of Situational Awareness activities are performed as part of a User Defined Operational Picture (UDOP) experience.



Tue Sep 28 17:09:16 EDT 2004

☐	?	multiple context alert test	CENTCOM	28 Sep 2004 18:31:13 GMT
☐	?	New DSIDE Workspace: CharlieTest	CENTCOM	23 Sep 2004 19:40:53 GMT
		Attachment has		

Property Name	Value
UUID	9DA54CE9-F25A-B855-26A1-D4D78F21D545
Name	53000PLAT
Threat	FRD
Category	UNK
Pen Color	🟦
Mil Code	SFSPXF--------N
Course (DEG)	38
Speed (KTS)	9.26
Lat Lon Alt	13:35:08N 165:40:10E

☐	Name	Category	Course (DEG)	Lat Lon Alt	Mil Code	Pen Color	Speed (KTS)	Threat	UUID
☐	GUMBY99	UNK				🟦		FRD	BF9BE358-CE10-7E79-444E-9BD3A374F254
☐	41000PLAT	UNK	38	77:45:55N 077:27:27W	SFSP--------N	🟦	33.336	FRD	FBD2C745-8AC6-F26B-D993-6379217CBB37
☐	OWN TRACK	UNK		78:21:13S 100:56:31W	SFSPO--------N	🟦		FRD	1D7DC30B-60D2-D76A-24AD-B7E4F3F820A7
☐	19000PLAT	UNK	43.21	62:04:32S	SFSPXF------	🟦	2.695	FRD	1BCA5961-DF94-8F14-

Figure 4: NCC Pilot Portal Application – User Viewpoint

One of the key operational threads within the definition of a UDOP involves the definition of a blue-force tracking "Operational Context". This is a task where the user selects from an available list of C2 blue-force data sets (available based on the user's profile), and customizes the mix of information that is to be extracted and subsequently viewed from the net-centric environment.

Figure 5 is a logical representation of what that operational thread looks like from a network, platform, and application standpoint. By logically grouping, and then modeling the components that support the user thread, we are able to determine the instrumentation necessary to derive what will constitute the health and well being of the business logic flow.

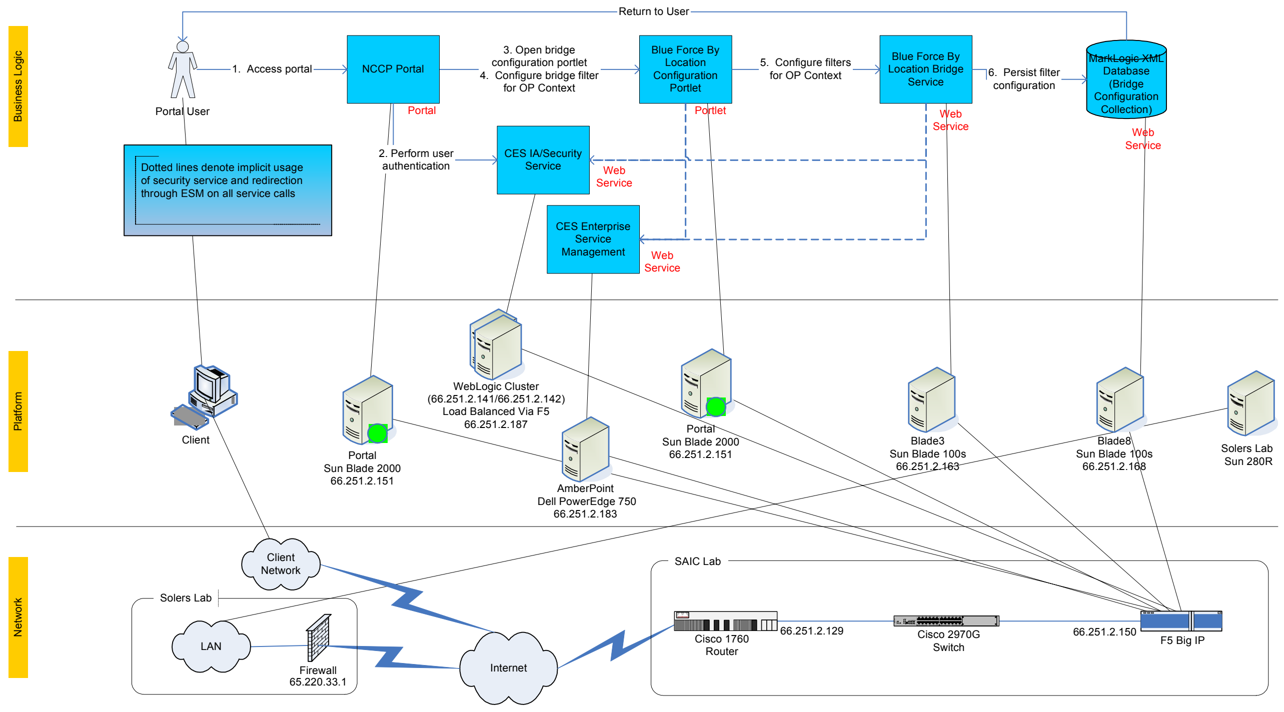


Figure 5: Logical Model: Blue Force Tracking—Operation Context Creation

Another way of viewing this type of model might be to envision it as the manifestation of a Web Services orchestration, where BPEL4WS has been converted (perhaps dynamically so) to a representation of its corresponding underpinning elements. Because it is a representation of the physical components, we label it a logical model from the enterprise management perspective.

4.3 Creation and Instrumentation of a Physical Model

Once the logical model is created, tools are required to instrument each of the layers and correlate them into a meaningful picture. The PEMS approach starts by choosing best of breed tools in each of the corresponding layers. As Figure 6 illustrates, many of these products have created and begun to embed Web Services interfaces to provide some transparency to their respective capabilities. The tools are installed and instrumented against all of the network, platform, or application entities that they correspondingly manage. For example, Tivoli is a popular platform management tool that can automatically discover computing platforms and manage CPU, memory, processes, disk drive, and other vital elements. That data can be exposed outward to third-party applications via the Tivoli Framework, which provides the hooks required to dynamically extract the data for upstream use.

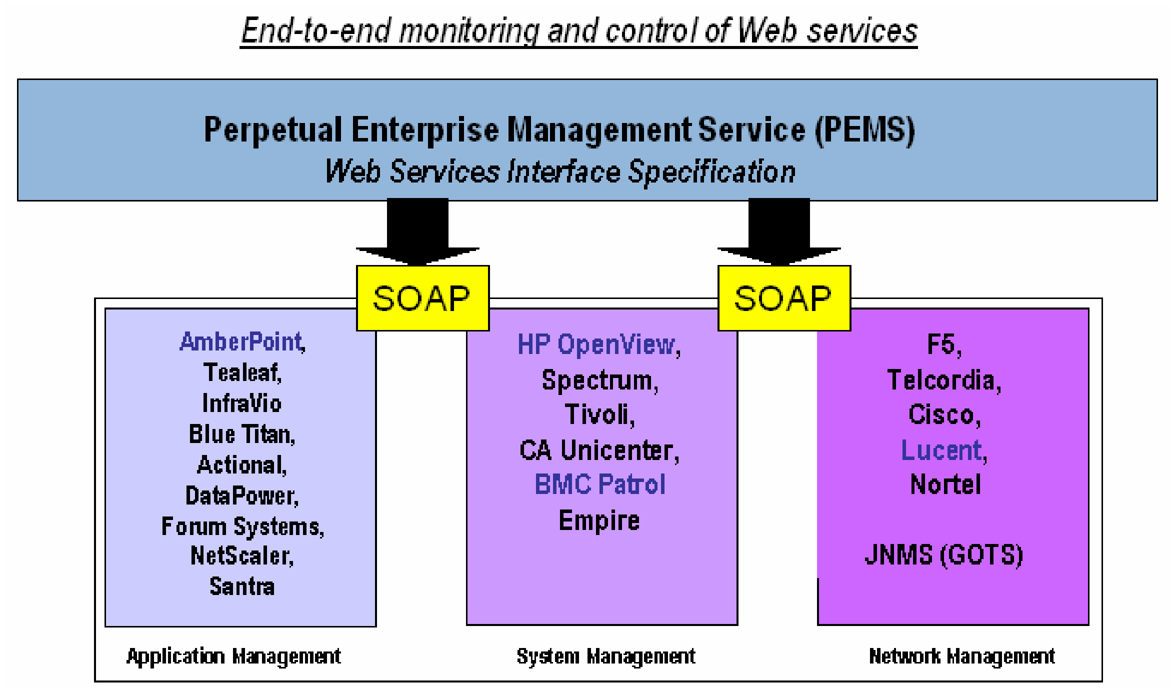

End-to-end monitoring and control of Web services
Perpetual Enterprise Management Service (PEMS)
Web Services Interface Specification
SOAP
SOAP
AmberPoint, Tealeaf, InfraVio Blue Titan, Actional, DataPower, Forum Systems, NetScaler, Santra
HP OpenView, Spectrum, Tivoli, CA Unicenter, BMC Patrol Empire
F5, Telcordia, Cisco, Lucent, Nortel
JNMS (GOTS)
Application Management
System Management
Network Management

Figure 6: Bridging Commercial Tool Suites into a PEMS Engine

A number of tools are starting to emerge in the commercial IT market space that enable aggregation and correlation of data from disparate sources into a common view. These "Manager-of-Managers" (MoM) as they are sometimes called, offer a method to create a physical representation, or model, of components that can then be coupled with an engine for extracting and assembling data that the model references. *Micromuse's NetCool* and *Lucent's ISA VitalSuite* are two such applications that perform this function. The PEMS engine shown in Figure 6 was built using these tools. A key factor in choosing the appropriate tool is the ability to re-construct the logical model in a meaningful way within the software.

Transforming the logical model into a physical model via a Commercial off-the-shelf (COTS) MoM tool is part art and part science. From a mechanical perspective, the objects that are represented in the logical model must be physically located (or discovered) on the network using a set of interfaces provided by the tool. This job can be simplified if strong interfaces between the MoM and underlying COTS tools that manage the application, platform, and network exist. For example, *Micromuse's NetCool* product contains a series of adapters that allow *Amberpoint, Tivoli*, and *Cisco* management tools to report up data into the *NetCool* correlating engine for processing. Likewise, if there is not a lower echelon management tool in place, some basic data can be extracted from the MoM itself. Finding the right combination and then representing it in a meaningful way for overall user thread monitoring purposes is part of the art involved in this process.

As Figure 7 illustrates, reconstructing the logical model swim lanes (application, platform and network) is best viewed in terms of a hierarchy, with an object representing the key user thread shown by an icon at the top of this heap. In our example, the 'Blue Force – Operational Context' user thread represented in this model is dependent on five key application constructs as shown directly beneath the icon at the topmost portion of the diagram. A Security Service, NCES Web Portal, Blue Force Configuration Portlet, MarkLogic Database, and Blue Force by Location Service are all primary application constructs for this user thread. These application elements in turn rely on a set of machinery and underlying network components that are further depicted in the diagram.

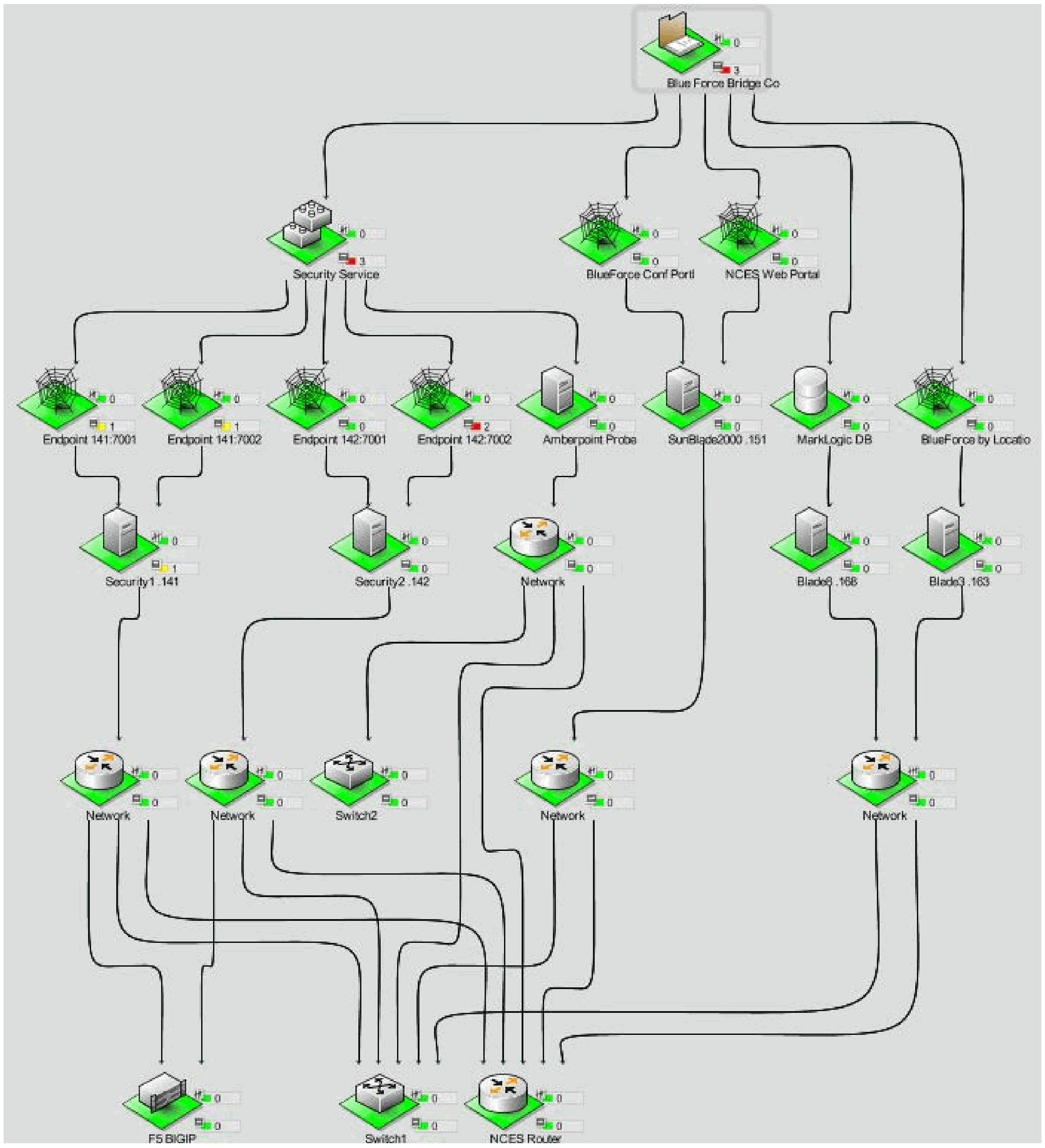
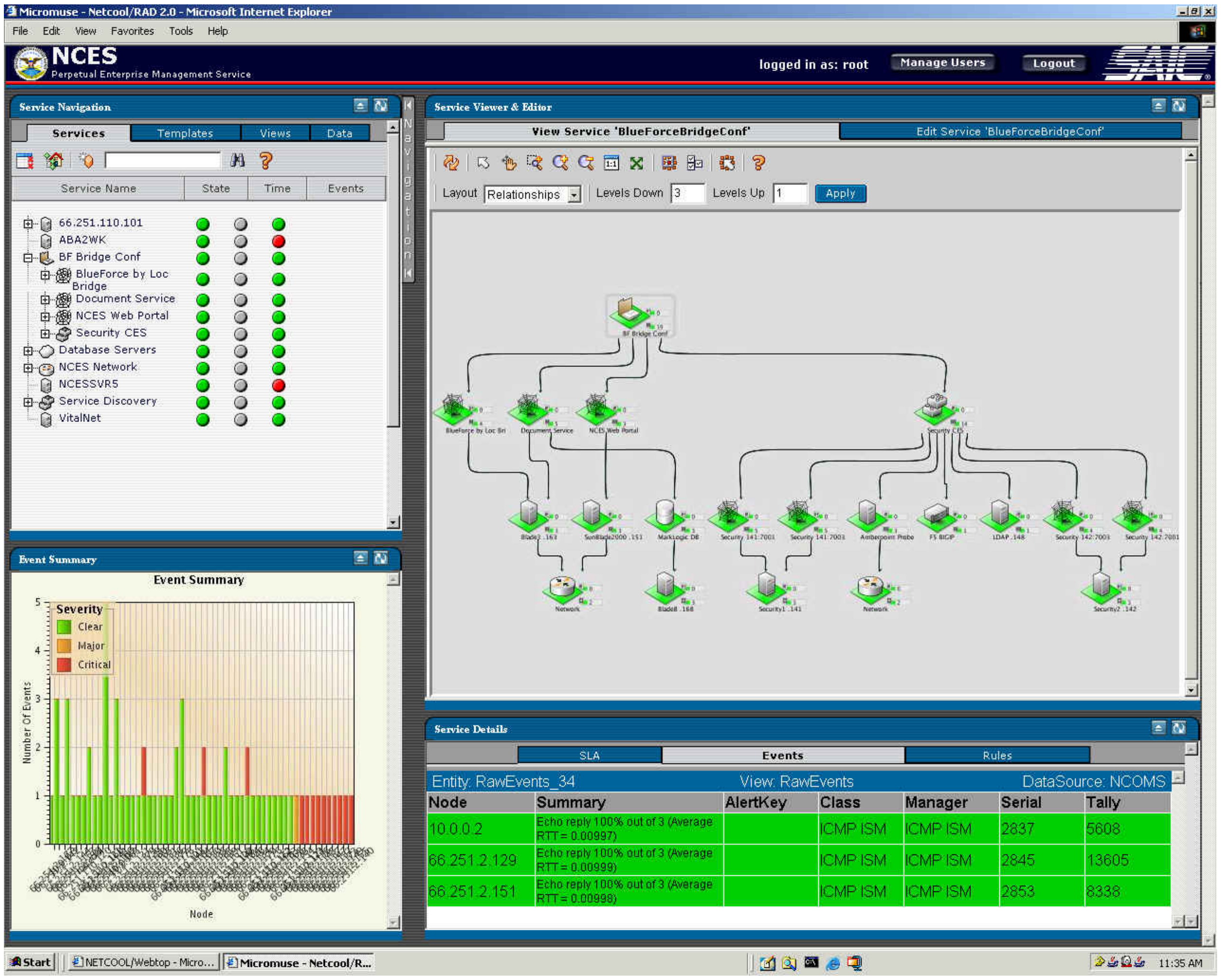


Figure 7: Physical Model: Blue Force Tracking—Operation Context Creation

One key item of note in this picture is the use of a load-balanced Security Service. As the diagram shows in the upper left portion, the Security Service is supported by two different systems, each running dual instances of the Security Service. Each system is balanced across an *F5 Big IP* device that separates the platforms into two networks. By studying this diagram one can identify all of the network, platform, and application pieces that are involved in the Blue Force - Operational Context user thread. The green squares that underscore each icon represent the current state of each element as the underlying management software is instrumenting it. *Tivoli* for example, is reporting on platform status, showing that according to its defined thresholds both the Security-1 and Security-2 systems are operating efficiently. *AmberPoint* (our Web Services Management tool) is reporting that the Web Services that support this thread are all operating as planned. *CISCO* management software is reporting that the network devices are up and operational. The instrumentation is being performed at the lower management tool level. That instrumentation (of platform, application, & network components) is sending data to *Micromuse NetCool* that aggregates and correlates this data into the picture we see in Figure 7. *NetCool* can further instrument the interconnections between these elements to establish what truly represents a SLA for the top-level Blue Force Tracking user thread or service.

By having a MoM tool that abstracts this data, the user thread can be monitored for health along side a number of key processes per the Thread Management Principle described earlier. Figure 8 illustrates how this can be represented in the PEMS model.



Layout: Relationships | Levels Down: 3 | Levels Up: 1

Service Name	State	Time	Events
66.251.110.101	🟢	⚪	🟢
ABA2WK	🟢	⚪	🔴
BF Bridge Conf	🟢	⚪	🟢
BlueForce by Loc Bridge	🟢	⚪	🟢
Document Service	🟢	⚪	🟢
NCES Web Portal	🟢	⚪	🟢
Security CES	🟢	⚪	🟢
Database Servers	🟢	⚪	🟢
NCES Network	🟢	⚪	🟢
NCESSVR5	🟢	⚪	🔴
Service Discovery	🟢	⚪	🟢
VitalNet	🟢	⚪	🟢

Entity: RawEvents_34 View: RawEvents DataSource: NCOMS

Node	Summary	AlertKey	Class	Manager	Serial	Tally
10.0.0.2	Echo reply 100% out of 3 (Average RTT = 0.00997)		ICMP ISM	ICMP ISM	2837	5608
66.251.2.129	Echo reply 100% out of 3 (Average RTT = 0.00999)		ICMP ISM	ICMP ISM	2845	13605
66.251.2.151	Echo reply 100% out of 3 (Average RTT = 0.00998)		ICMP ISM	ICMP ISM	2853	8338

Figure 8: PEMS Model showing User Thread Instrumentation

The diagram is split into four panes, providing a different level of granularity and aggregated detail in each window. This is how an administrator might manage an entire C2 application.

5. PEMS IN ACTION

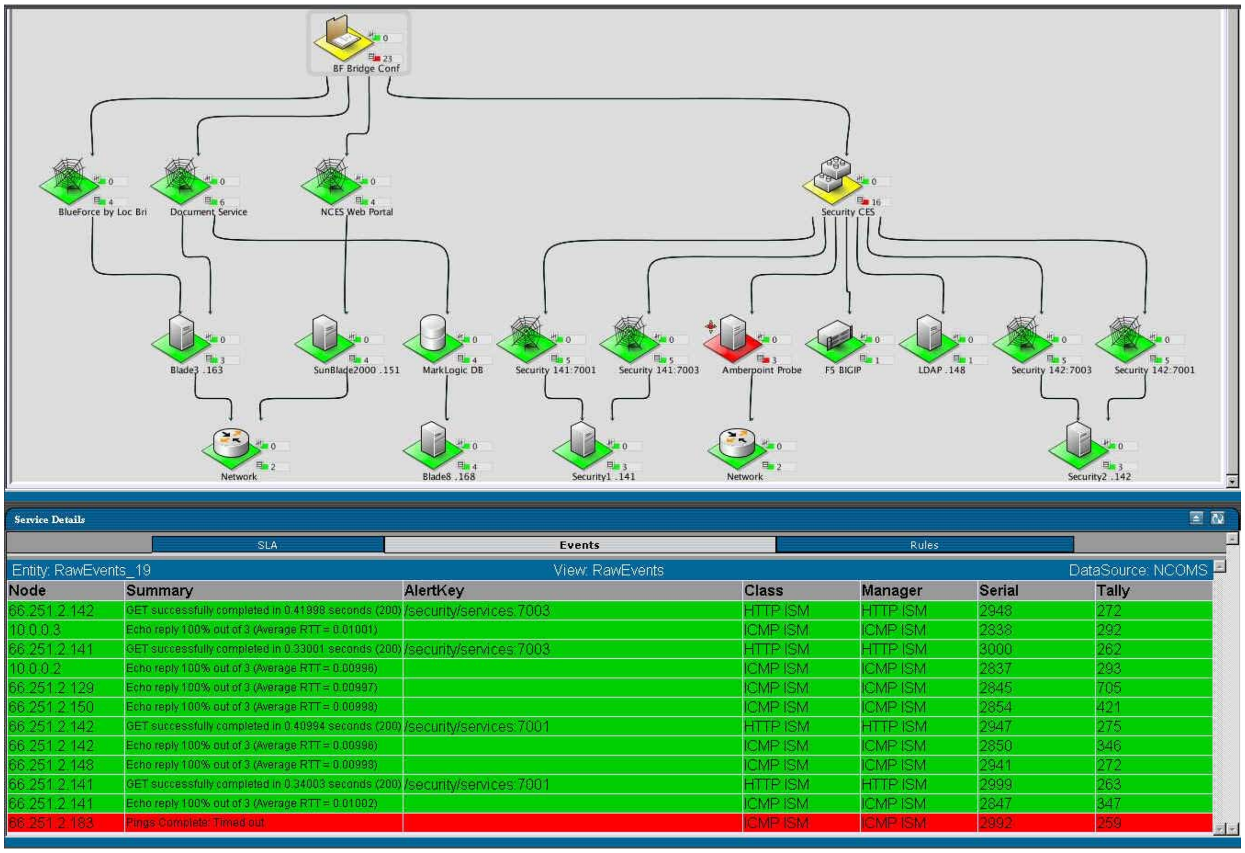
5.1 The Scenario

The best way to illustrate PEMS is via a simple example. Suppose the Blue Force user thread described in Figures 7 and 8 was somehow interrupted. For example, what if the system running our *AmberPoint* probe were to fail and simply crash? What would the upstream impact be of this issue? If we were just running our traditional tools we might discover that *Tivoli* indicates that a computing platform is down in the network. By itself, however, that would tell us nothing of the impact to the user. We may or may not get complaints from actual human users on performance or downtime. It is hard to say until the help desk phones begin to ring. Since it is the *AmberPoint* system that is down, we have effectively taken out our ability to report on our Web Services/app layer, so if we were counting on a pure Web Services Management approach, we would be left without any data to support the potential impact, if any, to the user experience.

Fortunately, the PEMS model gives us some additional insight into the problem by abstracting our view up from these management tools and allowing us to create a model that represents the interaction and relationships between key components.

5.2 PEMS Sense and Response

Figure 9 depicts how PEMS would represent our problem. As the diagram illustrates, there are three objects appear affected by some sort of anomaly that has occurred in our thread management process. The red square underling the *AmberPoint* platform brings us straight to the key problem. This box is down. We know it is down because the network supporting it is up, and other boxes on the network are up. Likewise, PEMS has placed a '+' symbol next to this computing system indicated that the source of the upstream issue starts here.



Entity: RawEvents_19 View: RawEvents DataSource: NCOMS

Node	Summary	AlertKey	Class	Manager	Serial	Tally
66.251.2.142	GET successfully completed in 0.41998 seconds (200)	/security/services:7003	HTTP ISM	HTTP ISM	2948	272
10.0.0.3	Echo reply 100% out of 3 (Average RTT = 0.01001)		ICMP ISM	ICMP ISM	2838	292
66.251.2.141	GET successfully completed in 0.33001 seconds (200)	/security/services:7003	HTTP ISM	HTTP ISM	3000	262
10.0.0.2	Echo reply 100% out of 3 (Average RTT = 0.00996)		ICMP ISM	ICMP ISM	2837	293
66.251.2.129	Echo reply 100% out of 3 (Average RTT = 0.00997)		ICMP ISM	ICMP ISM	2845	705
66.251.2.150	Echo reply 100% out of 3 (Average RTT = 0.00998)		ICMP ISM	ICMP ISM	2854	421
66.251.2.142	GET successfully completed in 0.40994 seconds (200)	/security/services:7001	HTTP ISM	HTTP ISM	2947	275
66.251.2.142	Echo reply 100% out of 3 (Average RTT = 0.00996)		ICMP ISM	ICMP ISM	2850	346
66.251.2.148	Echo reply 100% out of 3 (Average RTT = 0.00998)		ICMP ISM	ICMP ISM	2941	272
66.251.2.141	GET successfully completed in 0.34003 seconds (200)	/security/services:7001	HTTP ISM	HTTP ISM	2999	263
66.251.2.141	Echo reply 100% out of 3 (Average RTT = 0.01002)		ICMP ISM	ICMP ISM	2847	347
66.251.2.183	Pings Complete: Timed out		ICMP ISM	ICMP ISM	2992	259

Figure 9: PEMS View of Problem and its Impact to a Key User Thread

The Service Administrator can then look at how this impacts the upstream thread. We see that the *AmberPoint* system is only being used to actively monitor (via agent software) the Security Service that is used to authenticate transactions against our Blue Force user thread. The *AmberPoint* agent software is likely impacting the performance of the security service because it is not able to communicate back to its host system. While there is a performance hit, it does not appear to be severe enough (as indicated by the 'yellow' status square) to break the Security Service. Likewise, it is indirectly having an impact on our top-most thread, but not enough to bring down our overall user experience. All of these elements are reporting status, and by viewing that status against an instrumented set of parameters that is aggregated up into a single operational picture, we are able to trace through the issues quickly.

One key feature of PEMS is that a response can be quickly engaged to fix this problem. The panel in Figure 9 has dynamic hooks into the lower level management tools that support it, allowing us to drill down and examine the issue in further detail. Likewise, status data is provided through a series of messages reported beneath our physical model diagram. The combination of additional tool access directly via the symbols represented on the diagram, coupled with interrelationships presented on screen with their direct status, allows a Service Administrator to quickly assess, and determine a required response.

6. CONCLUSION

The advancement of SOA-based application development and deployment has given rise to a new realm of I.T. management complexity. While the commercial marketplace has focused on delivering software and standards that have helped speed the production of net-centric capability, the tools needed to support dynamic management of this new form of distributed computing are still evolving. By leveraging a mixture of existing network, platform, and application management software with a new form of manager-of-manager tools, SOA user threads can be modeled and managed in ways that have not previously been considered.

If the Thread Management Theory described in this document holds true, implementing a Perpetual Enterprise Management Service is possible by combining these new tools with the process of ‘user thread’ logical and physical modeling. A new role called the Service Administrator will emerge as a key figure, combining and coordinating the skills of network, systems, and software engineers to derive meaning from these models, and respond to SOA management issues in a far more efficient manner than is currently being deployed under traditional client/server or pure Web Services Management approaches.

This paper has outlined a set of ongoing research and development that is exploring this problem in depth, looking to optimize how complex problems presented by the boundless nature of next generation C2 applications (built on an SOA fabric) can be effectively managed and supported in an operational DoD environment. The objective of this research is to bring to light new and innovative ways to help manage the complexity of C2 systems of the future.

This page intentionally left blank.

7. BIBLIOGRAPHY

1. WILL IVERSON (2004). “Web Services in action: Integrating with the eBay Marketplace”, *O’Reilly publishing*, page 7.

2. CHRIS PELTZ (2003). “Web Services Orchestration: A review of emerging technology, tools, and standards”, *Hewlett Packard Press*, page 6.

3. EBAY, Inc. (2004). API Technical Documentation (Version 399), “Basic Building Blocks” located at: http://developer.ebay.com/DevZone/docs/API_Doc/index.asp

This page intentionally left blank.

8. ACRONYMS AND ABBREVIATIONS

BPEL4WS	Business Processing Execution Language for Web Services
C2	Command and Control
COTS	Commercial of the Shelf
CPU	Central Processing Unit
DISA	Defense Information Systems Agency
DoD	Department of Defense
FCS	Future Combat System
GIG	Global Information Grid
GOTS	Government off-the-shelf
IT	Information Technology
JC2	Joint C2
JNMS	Joint Network Management System
LAN	Local Area Network
MCP	Mission Capability Package
MoM	Manager of Managers
NCCP	Net-Centric Capabilities Pilot
NCES	Net-Centric Enterprise Services
NCW	Net-Centric Warfare
NGC2	Next Generation Command & Control
PEMS	Perpetual Enterprise Management Service
SLA	Service Level Agreements
SLO	Service Level Objectives
SOA	Service Oriented Architecture
UDOP	User Defined Operational Picture
WAN	Wide Area Network
WSDL	Web Services Definition Language
WSM	Web Services Management
XML	eXtensible Markup Language

This page intentionally left blank.