USE OF PRODUCT LINE ARCHITECTURE FOR MULTI-USE SIMULATIONS

Lawrence A. Rieger
U.S. Army Training and Doctrine Command
TRADOC Project Office One Semi-Automated Forces
Fort Monroe, VA 23651

Cynthia T. Harrison
U.S. Army Simulation, Training, and Instrumentation Command
Product Manager, One Semi-Automated Forces
Orlando, FL 32836

ABSTRACT

The One Semi-Automated Force (OneSAF) program is being developed to replace a number of models and simulations currently used in the training domain as well as those used for analytical purposes. With the Operational Requirements Documents requiring both human in the loop and closed form, stand alone and distributed modes of operation, and support to a myriad of users, a single monolithic simulation could not satisfy the variations in expected uses. Use of an innovative product line architecture which allows the user to compose a particular instantiation of the simulation for each use case from a hand basket of tools, will enable OneSAF to meet it's widely varying user needs. The product line architecture will also be designed for constant replacement or modification of tools and modules as dictated by the evolving tactical operational requirement. The paper details the breadth of the requirements, and differing operational architectures, and how the architecture is designed to be instantiated from multiple tools and models, by the user, at runtime.

AUTHOR BIOGRAPHIES

Mr. Lawrence A. Rieger is the Technical Advisor to the TRADOC Project Office-OneSAF, Deputy Chief of Staff for Training and Operations, HQ TRADOC, U.S. Army. He received a BA from Belmont Abbey College in 1976 and an MS from Troy State University in 1982. He is also a graduate of the Army Command and General Staff College and the Army Management Staff College. Following active and reserve commissioned service with both light and mechanized forces, he has spent the last 17 years in the development and management of simulations for training, working in the live, virtual, and constructive simulation environments.

Ms. Cynthia T. Harrison currently works for the U.S. Army as Project Director for the OneSAF program. Since 1985, she has had a wide variety of experience in the Live, Constructive, and Virtual simulation domains for the Army and Navy. Ms. Harrison holds Bachelor and Master of Science degrees in Computer Science from UCF.

Use of Product Line Architecture for multi-use simulations

Lawrence A. Rieger
U.S. Army Training and Doctrine Command
TRADOC Project Office One Semi-Automated Forces
Fort Monroe, VA 23651

Cynthia T. Harrison
U.S. Army Simulation, Training, and Instrumentation Command
Product Manager, One Semi-Automated Forces
Orlando, FL 32836

INTRODUCTION

The One Semi-Automated Forces (OneSAF) is the next generation semi-automated and computer-generated force being developed to meet the total Army requirement for entity-level training and analytical modeling and simulations. OneSAF is defined as A Composable next generation Computer Generated Force (CGF) that can represent a full range of operations, systems, and control processes (TTP) from entity up to battalion level, with variable level of fidelity that supports all M&S domain (ACR, RDA, TEMO) applications with a emphasis on human-in-the-loop and no human-in-the-loop. [1] The need to satisfy cross-domain requirements provided the Army with a massive challenge, since all three domains have differing M&S requirements. The domains are; Advanced Concepts and Requirements (ACR) — requiring a set of analytical tools which yield combat outcome data in sufficient detail to provide the analyst with necessary information on which he/she can make supportable judgments as to why things happen. Research Development and Acquisition (RDA) — requiring a capability to place a weapons system within a realistic battlefield and make judgments as to its relative worth over another similar competing system. And Training, Exercises and Military Operations (TEMO) — requiring both an environment in which to enhance soldiers skills by presenting them with a realistic setting in which to practice and enhance those skills, and a means to plan, evaluate, enhance and rehearse operational courses of action on real-world terrain data bases against known and suspected opposing forces and conditions. These three domains traditionally use models and simulations having differing levels of fidelity and different means of managing time.

BACKGROUND

Historically, simulations have been developed for a specific application within a specific modeling and simulation domain. They have been designed and developed for this particular application and set of user conditions, and are modified at usually significant expense and effort when a major change is required, or are replaced when it is uneconomical to upgrade to contemporary needs. A major part of this process has been that simulation technology and capabilities continue to develop at a significant rate, and most fielded simulations have been limited in functionality and growth potential by the technologies available when they were initially developed. When the OneSAF program was initiated, the Army was directed to provide a simulation that really was the first of it s kind. It was to be software only program capable of operating on a plethora of hardware platforms. It was to be used by all three simulation domains, which meant it had to have various time management schema and had to be both closed loop and human in the loop. And it was required to be a both a composable simulation and a scalable one. When these requirements were all added together, the task seemed impossible, and in fact, an independent research effort said the requirement could not be met by any existing product, or known architecture technology[2]. Since the Army lacked the resources to build multiple new SAFs to modernize its legacy M&S array, and modernization was urgently required, the only solution was to develop a new technical approach to simulation architecture.

TRADITIONAL SIMULATION ARCHITECTURE

Traditional simulation architectures are monolithic in nature and are frequently limited to operation on a specific hardware platform. What this means is the simulation software code is designed for the interfaces and services of a specific computer platform and operating system. It also means that the software is designed to operate in a fixed and determined process, on that particular hardware and operating system, and requires complete recompiling if changes are made. In practice, this means that there is a very fixed approach to modifying and enhancing the software code, including the addition of new models and behaviors, which results in specific blocks or versions of the simulation. It also very commonly meant that the initial developer virtually owned the process for improvements, and became an expensive sole source provider of those upgrades.

A PRODUCT LINE ARCHITECTURE CONCEPT

Before detailing the OneSAF Product Line Architecture, it is appropriate to examine the requirements that drove the architectural solution during development. OneSAF is required to operate in all three military simulation domains, ACR, RDA, and TEMO, each of which have different needs, and each of which have very different typical simulation operating modes. In the detailing of user requirements, the three domains provided a total of 44 distinct use cases, each requiring a different version of the hardware host and software functionalities or time management schema. To understand this more completely, a review of the different user domains is appropriate.

The ACR domain focuses on determining the need for new equipment or changes in Army organization or tactics. The typical simulation operates on a single platform, using moderate to high fidelity models, and operates in discrete event time management schema to ensure repeatability of the model runs. The RDA domain focuses on the development aspects of new equipment and will typically operate with some extremely high fidelity engineering models linked to a synthetic natural environment and Computer Generated Forces behavior models. RDA simulations generally operate at an extremely fine time step or discrete event step to determine and record what is happening at the engineering level of events. The TEMO domain frequently operates in one of several common modes: 1) A real time SAF tied to manned virtual simulations, 2) A time step or discrete event distributed simulation used as a staff exercise driver, and 3) A stand alone course of action development and analysis tool utilizing as fast as possible time management. These modes typically require a variety of hardware configurations consisting of multiple interacting machines, central servers with peripherals, and stand alone machines.

Given this wide spread of requirements, especially the various time management and hardware configurations, a traditional monolithic approach to architecture design was deemed impractical, with massive technical and cost risks. Another major technical risk was the requirement from Army leadership that the software had to be platform independent to the greatest degree possible. This requirement had never been levied on a major Army simulation program prior to OneSAF.

These requirements drove the OneSAF team to develop a new architecture concept, a composable architecture consisting of a framework structure, common services, components and products. Defined as a Product Line Architecture (PLA), it groups necessary simulation components in one architecture layer, combines components into products at the next level, and then assembles those products, and necessary services, into an instantiation of OneSAF for a particular user application. Each instantiation of the PLA consists of various products, themselves composed of components and supporting services, which are what the user needs for his application and nothing more. Thus each user application or instantiation is potentially a unique configuration of the PLA.

THE PRODUCT LINE ARCHITECTURE FRAMEWORK

The heart of the OneSAF architectural approach is the Product Line Architecture Framework (PLAF) Figure 1,[3] which provides a mechanism to organize, categorize, and define the layered software structure in order to incrementally meet the OneSAF requirements. The PLAF provides a static view of the system configurations, products, and components that make up the OneSAF PLA and identifies the software products and components that can be used as building blocks for composing higher level functionality. For an example, looking at the Component Layer under the Model Composer product layer, the user may use any or all of the Entity, Unit, Behavior or Environment components to form the Model Composer product he/she needs for that application. In several of the TEMO use case applications, the user may not even create any model composer product within that instantiated architecture as hey have no nee to modify the existing validated unit/entity/behavior compositions. The TEMO user may, however, include the C4I Adaptor product as they may have a requirement to interoperate with organic C4I devices. Alternately, an ACR user developing tactical doctrine or analyzing alternative force structures is likely to use all four of the model development components within that application architecture.

The PLAF provides the structure within which the user composes the specific tools, services and products required for their particular application, and also enables the user to eliminate from the larger OneSAF architecture toolbox the tools that are not needed for their use. This ability to remove unnecessary components and products enables the simulation to operate more efficiently, improving the runtime performance, and reducing the amount of simulation overhead that slows down performance on any given hardware platform.

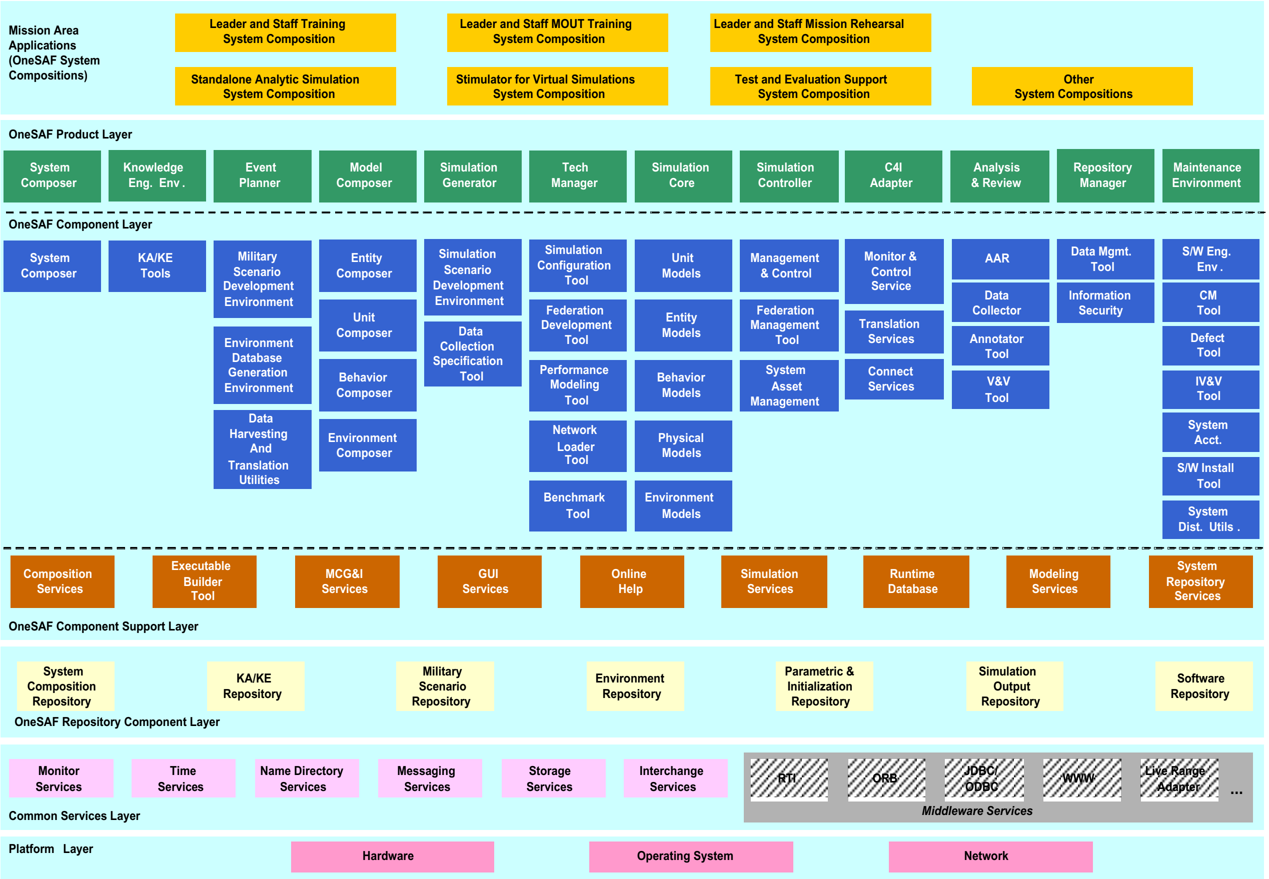

Figure 1: OneSAF Product Line Architecture Framework

A PRODUCT LINE DEVELOPMENT STRUCTURE

Before we consider a detailed discussion of the PLAF instantiations for the various domain use cases, a detailed look at how the product line works is appropriate. The most critical issue in a product line architecture is ensuring that all components and products work together, and the simulation instantiation is readily adaptable to working with other simulations and models. In the traditional monolithic simulation, this is not a substantial issue, as the software is designed and developed to run as one massive block program. However, by definition, a product line architecture is a group of autonomous components, products and services operating together within a defined framework. This demands that the capability of various components, products and services to interoperate at runtime be addressed as a critical design issue. For the OneSAF program, both a detailed interface specification and a management process are utilized to prevent component incompatibility.

To support the management process, a single contractor was selected to be responsible for the specification and development of the overall product line architecture and the integration of all components into the product line. The components are developed by a variety of contractor teams co-located with the Architecture and Integration contactor in a collaborative development environment.

The component software is developed within a series of builds within each major product development block and must meet a component contract as specified by the PLA. At each build, as well as for the entire block, a detailed integration testing process is followed, ensuring that all of the various services, components and products meet the published component contracts, and that the individual components and services do not generate a runtime incompatibility despite meeting the interface specification. This may not sound all that unusual, except that different groups of components; models, products, services and the environmental representation, are being developed by different contractors, and are being continuously integrated into the developing product baseline.

The component contracts are part of the Product Line Architecture Specification (PLAS), which is very detailed, yet constantly evolving through the development process. The primary contractor developed a PLAS at the origin of the development effort, and in coordination with all contractors, continues to revise and update the PLAS as the various components and services are more completely detailed and defined.

The combination of both the detailed and evolving PLAS and the constant integration and test process of each build of code ensures that, regardless of the composition or host platform, each instantiation of OneSAF will operate without component or service incompatibility or failure.

ONESAF PROGRAM DEVELOPMENT

OneSAF is being developed and fielded in a series of major development blocks. Unusual for a military simulation program, early blocks of functionality will be provided to selected users for feedback as well as general use. As described above, each software block undergoes substantial integration tests to ensure both service and component functionality and interoperability with the architecture framework, services, and other components. Each block is an expansion of earlier developed functionality, and is not a replacement of previous software. The letter coded functionality blocks are initially A-D, with the intent that additional blocks will follow on as new functionalities are added as requirements, and program funding is available to meet P3I capabilities. The following highlights the primary capabilities required for each Block delivery;

Block A focuses on specification and development of the PLA. It provides a working architecture framework and toolset to support development of modeling capabilities in future blocks and is available 4th Qtr, FY02.

Block B provides the first Military Useful Functionality with the minimum capabilities equal to JANUS and JCATS(MOUT). Block B is the first fully functional block available to selected users, and is scheduled for completion 4th Qtr, FY03.

Block C provides a capability equivalent to the OneSAF Testbed Baseline (OTB) and is the initial functionality projected for general user release in the 4th Qtr, FY04.

Block D constitutes the Full Operational Capability and contains all threshold requirement functionalities in the OneSAF ORD with an expected release 4th Qtr, FY05.

Block E and beyond are for Pre-Planned Product Improvements and will be developed and fielded as requirements are prioritized and resources become available. We assume an annual release of configuration managed enhancements and improvements.

As can be seen by this development program, at roughly yearly intervals, various components and services will be updated and augmented, and new functionalities will be added to the fielded version of OneSAF. This is one of the significant advantages of the Product Line Architecture, that individual tools, services, and components can be either updated or replaced, in the development cycle or after operational fielding, without requiring a complete re-write of the software code, merely replacement of the software code module that provides that particular functionality. Although this is similar in concept to modular software design, a key difference is that each module of code is a free-standing function, working through a defined interface specification, rather that a module of code existing within a monolithic program. The system architects believe that this approach will prevent the incompatibility glitches that have so plagued other simulation programs when upgrades or refinements have been added to an existing simulation software package.

This building block approach was selected as a reflection both of the unusual Product Line Architecture approach of the program, and to provide usable product to the Army for limited early release and user feedback. This is also unusual, as previous simulation programs traditionally took the requirements document and would spend many years in development, then drop a completely unknown product into the field. The OneSAF building block approach will sequentially field capabilities enabling early retirement of the OneSAF Testbed baseline (which has replaced ModSAF in Army service) and functionalities enabling other systems to be replaced or retired prior to Full Operating Capability in 2005. The building block approach is also an excellent vehicle for the application of the eXtreme Programming form of software development utilized by OneSAF.

COMPOSED ARCHITECTURES FOR DOMAIN REQUIREMENTS

This paper details four different notional system compositions for OneSAF, which cover the vast majority of user requirements. While each notional composition will have numerous variations by their different users, they do demonstrate how the differing compositions will meet user needs, and the value of a Product Line Architecture in meeting multiple user requirements. These four compositions are; 1) An ACR/RDA domain single platform stand alone, 2) An RDA domain single platform federation with another model, 3) A TEMO domain multi-platform distributed training staff exercise, and 4) A TEMO domain single platform C4I exercise driver. All four instantiations follow the Software Implementation Layering Diagram[4] shown in figure 2, which is the common generic structure of a OneSAF instantiation. Fielded versions of the system will include an composition wizard similar to existing PC installations wizards to guide the user through the composition process. It is also anticipated that several user-selectable default compositions will be pre-loaded in the wizard, enabling users to select a basic domain composition, and perform minor variations to make a fit for the specific user need. All the notional architectures will use several common tools and functionalities, the most common of which are the Military Scenario Development Environment (MSDE) used to develop the military scenario and the Data Collector and After Action Review (AAR) tools for data collection and manipulation.

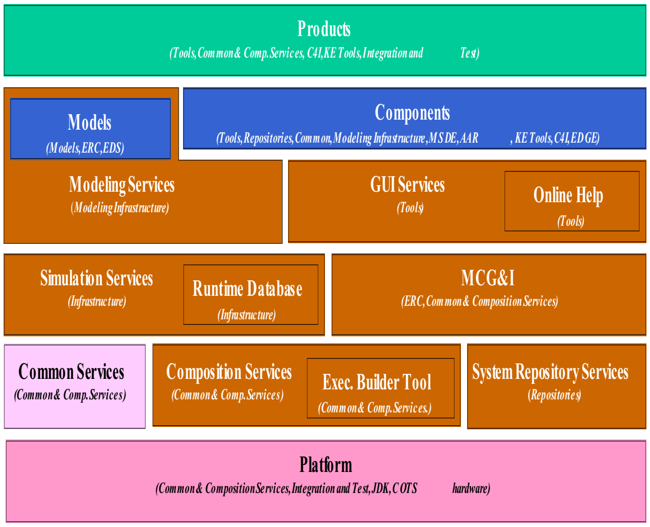
Products
(Tools, Common & Comp. Services, C4I, KE Tools, Integration and Test)
Models
(Models, ERC, EDS)
Components
(Tools, Repositories, Common, Modeling Infrastructure, MSDE, AAR , KE Tools, C4I, EDGE)
Modeling Services
(Modeling Infrastructure)
GUI Services
(Tools)
Online Help
(Tools)
Simulation Services
(Infrastructure)
Runtime Database
(Infrastructure)
MCG&I
(ERC, Common & Composition Services)
Common Services
(Common & Comp. Services)
Composition Services
(Common & Comp. Services)
Exec. Builder Tool
(Common & Comp. Services.)
System Repository Services
(Repositories)
Platform
(Common & Composition Services, Integration and Test, JDK, COTS hardware)

Figure 2: Software Implementation Layering Diagram

The ACR/RDA stand alone example composition is based on a workstation hardware platform, a SUN platform being used for current development testing. The user will typically select and instantiate unit, entity, environment and behavior composers, utilize conservative Discrete Event Simulation for time management, and single-entry random number table for exercise repeatability. The system is then instantiated, the user selects and modifies a standard scenario from library, composes, modifies or imports the entity, unit and behavior models necessary for the scenario, and selects the data collection elements desired. This application is commonly run in closed loop mode, without human intervention once exercise is initiated, and will commonly be run numerous times for both repeatability and replicability of exercise date.

The RDA federation example composition is based on a LINUX-based high end desktop PC running an instance of OneSAF federated with a workstation hosting a high fidelity engineering model used to determine proposed or prototype system configuration or design. The user will typically require the environment composer to create the necessary physical environment necessary to support test perimeters, and will use time management slaved to the high fidelity test model with which it is federated. The federation interface is expected to be through the use of the High Level Architecture (HLA), however, the OneSAF instantiation can also support federation interaction s using Distributed Interactive Simulation (IEEE 1278) Protocol Data Units (DIS PDU). Time management is commonly HLA time management services, but OneSAF time management may also be used, with the federated model being the critical discrete event controller. This instantiation would commonly use some portion of a second to be the discrete event.

The TEMO distributed training exercise instantiation is one of the most challenging to the OneSAF program. An intricate version of this use case could require a brigade headquarters server linked with four or five maneuver and support battalion servers, all driving either a network of user interface machines (desktop PCs) or individual Tactical Operation Center (TOC) Local Area Networks (LAN) to drive operational C4I devices by using the C4I Adaptor Product. This instantiation is not expected to use any of the composition tools, as they are likely to use an already composed and validated set of models to support the training objective. Time management remains discrete event simulation, but the discrete event is a specified time interval, notionally one second, and is controlled to be one to one ratio to the exercise wall clock, therefore operating in effective real time. Unlike the previous two instantiations, the TEMO use case has humans in the loop as both primary/secondary training audience and as role players and technical controllers. Host machines are expected to be either workstations using UNIX or workstation equivalent PCs using LINUX or WINDOWS, while the various interface machines would be common desktop PCs using WINDOWS or equivalent. It is also expected that additional machines, of workstation strength, will exist for exercise controllers, driving 3D stealth views or Plan View Display (PVD) graphics, and being used for data collection and preparation purposes.

The last common instantiation is the TEMO C4I exercise driver, which will use a single workstation strength PC, using LINUX or WINDOWS, to stimulate the operational C4I systems of an operational unit, most commonly expected to be a combined arms battalion, being directly connected to the TOC LAN. Very few of the tools and components from the user toolbox are commonly used, and quite probably, the user will do little other than instantiate the architecture using a previously prepared or modified file from the MSDE tool. This TEMO application, like the previous, will operate in effective real time.

ADVANTAGES TO A PRODUCT LINE ARCHITECTURE

All four of the user applications described above have one primary attribute in common. None of them used all of the software tools and components available within the OneSAF product line architecture. And one of them, the TEMO C4I driver, used very little of the available component tools and editors in the OneSAF toolbox. Why is this important to a simulation user? The instantiated software package uses all of the code that is made part of the runtime architecture, and does not waste computational cycles reading, then discarding, software code excess to the users needs. This means the simulation runs much more effectively and efficiently on the host machine, and the user does not pay for hardware capability he does not require for his application.

Another significant advantage of the product line architecture is that any module can be replaced by another when the user needs it, without the need to create a new simulation. Modules provide specified functionality to the composed instantiation. If the default stock tool or component does not meet the user need, or he has another tool more appropriate to his needs, then the architecture framework can accept the new module as long as the architecture API specification is adhered to. In this way, the PLAF can accept whatever tools and components, or additional functionalities, are required by the user.

The last significant advantage to the Product Line Architecture is that the program maintains not only a constantly modernizing toolbox of components and tools, negating the constant threat of obsolescence, but the PLAF and PLAS, by their very existence, encourage users to improve the simulation through developing new models and components, and submit those new developments for testing and incorporation into a constantly improving user baseline. There are two parts of this. The Army, and all the services, are constantly developing new pieces of equipment, and changing tactics, doctrine and organization to meet the changing world. The OneSAF toolkit enables the combat developers, using organic tools, to develop the entity physical and behavior models needed by users in the field. These new models are submitted back through the Configuration Control Board, tested for API compliance and vetted for appropriateness, and are then added to the next fielded version of the OneSAF baseline. And even when these models are experimental or unapproved, compliance to the standard API ensures they can be used by and shared between users, even if never added to the baseline. The second part of this is that individual modules, whether components or complete tools, can be replaced when a better tool is developed, or when the original module no longer meets user needs. It also allows for multiple modules to be used for the same required application. The OneSAF PLAF can contain several AAR tools, each optimized for a particular user application, and the end user can merely select the one that best suits the application need. Yet the architecture framework remains unchanged, and the user instantiated application software code is no larger than that absolutely necessary for the end use.

Select only what you need, and pay no parasitic loss of hardware processing capability for unneeded software blocks.

SUMMARY

The use of a product line architecture, a framework with specified interfaces and common service components, enables developers to provide a very large toolbox that can meet all user needs, and yet still provide the user with a runtime simulation that contains only the software tools and services he needs. The software can be continuously updated and improved without monolithic code re-writes, and various functionalities and components can be added, or removed, as the program needs demand.

[1] TPO OneSAF Command Briefing, presented to Naval Postgraduate School, 1 Aug 2001

[2] SAIC, OneSAF Technical Assessment, 12 Sept, 1997, SAIC-98/7765&00

[3] Anthony.J. Courtemanche, Robert L. Wittman Jr; OneSAF: A Product Line Approach for a Next-Generation CGF, Proceedings of the 11th Conference on Computer Generated Forces.

[4] Robert L. Wittman Jr, Anthony.J. Courtemanche, The OneSAF Product Line Architecture: An Overview of the Products and Process, proceedings of the May 2002 SimTecT