Osseus, An Experiment in Next Generation LVC M&S Architecture

Mark Riecken
Trideum Corporation
Orlando, Florida
mriecken@trideum.com

Derrick Franceschini
StackFrame, LLC
Sanford, Florida
derrick@stackframe.com

Scott Gallant
Effective Applications
Orlando, Florida
scott@EffectiveApplications.com

John Rutledge
Trideum Corporation
Huntsville, Alabama
jrutledge@trideum.com

Walter Barge
Director, Joint Assessment and Enabling Capability (JAEC)
OSD Force Readiness and Training (FRT)
Alexandria, Virginia
walter.s.barge.civ@mail.mil

ABSTRACT

Live, Virtual, and Constructive (LVC) technologies provide a powerful range of capabilities that the distributed training community enjoys in support of its broad spectrum of training needs. The fundamentals of this LVC capability were built over two decades ago at a time when the modeling and simulation (M&S) community was arguably ahead of commercial information systems, especially in terms of distributed computing and networking. With the advent of many web-based technologies the LVC community now finds itself attempting to integrate new technologies with legacy architectures. Due to the flexibility of the new technologies as well as the inventiveness of the LVC community, this approach has had some success, but these approaches continue to require specialized skillsets and can be costly to establish and maintain. The Defense M&S Coordination Office (DMSCO) has sponsored an effort to describe and prototype selected features of a next generation architecture that leverages recent and emerging technology more directly. This paper describes a framework called Osseus to accomplish these goals. Osseus incorporates desired next generation characteristics such as 1) more open and flexible interoperability between disparate systems; 2) the ability for relatively untrained users to fill in functionality gaps between available systems by dynamically injecting behavioral changes into the LVC environment; 3) the ability to connect services with granularity smaller than an application to increase the capability of the environment; 4) a more accessible means of composing distributed training capabilities for an educated, but non-specialist trainer; 5) data filtering to optimize or reduce data transmission over the network; and 6) centralized data management to facilitate tools such as visualization, data collection, and analysis. This paper discusses architectural aspects of Osseus and selected prototype results which include the integration of OneSAF with an example virtual system.

ABOUT THE AUTHORS

Mark Riecken Dr. Mark Riecken has over 30 years of experience in modeling and simulation (M&S), command and control (C2), mission command (MC), information technology and systems engineering and architecture. He has served as chief engineer on both large and small M&S projects and is currently chief engineer at the Trideum Corporation. Dr. Riecken has provided engineering and analysis expertise to many major M&S programs from the early days of Advanced Distributed Simulation Technology (ADST) to current support to DMSCO and other M&S organizations. Dr. Riecken is an architect in the Simulation to Mission Command Interoperability (SIMCI) Overarching Integrated Product Team (OIPT). Dr. Riecken holds a Ph.D. in electrical engineering from the University of New Mexico (1988), an M.S. in physics from Vanderbilt University, and a B.S. in physics from the University of Evansville (Indiana).

Derrick Franceschini Mr. Franceschini is Vice President and Senior Systems Architect at StackFrame, LLC. Mr. Franceschini has over 18 years' experience in the M&S industry. He has extensive experience in the Constructive and Virtual domains and is sought after in the industry for his expertise with architectural innovations for the U.S. Army's OneSAF Product Line. He is the leading force behind the conceptualization and development of the WebMSDE, a web-based Scenario Development Environment, WebSAF, a thin client, web browser-based application to visualize and control the Army's OneSAF Simulation, WebAAR, a modern, efficient web browser based After Action Review (AAR) system and the MobileSimCenter, a set of innovative products to provide easily deployable Constructive simulation capabilities. Mr. Franceschini served as the OneSAF System Architect from 2006 – 2010 and has held key development positions on the program since 2001. Earlier in his career, he served as the Lead Engineer on the Modular SAF (ModSAF) and OneSAF Testbed Baseline programs. He received his B.S. in Computer Engineering from UCF.

Scott Gallant Mr. Gallant is a Systems Architect with Effective Applications Corporation. He has 20 years' experience in distributed computing including United States Army Modeling & Simulation (M&S). Scott has led technical teams on distributed M&S programs for distributed software and federation design, development and execution management in support of technical assessments, data analysis and experimentation. He currently leads the technical team for the implementation of the Executable Architecture Systems Engineering (EASE) system and actively supports research activities of the Army Research Laboratory, Human Research and Engineering Directorate, Simulation and Training Technology Center (ARL HRED STTC) Advanced Simulation Branch.

John Rutledge Mr. Rutledge is a deputy Director in the Trideum Corporation.

Walter Barge Dr. Walter Barge is a Department of Defense employee working in the Office of the Secretary of Defense (OSD). Shep's primary focus is in the areas of assessment and policy formulation for defense readiness and training. This work involves data analysis and visualization, performance measurement, and enterprise-level assessments of Joint training activities. His background is in Operations Research and one of his current interests is knowledge management practices within DoD.

Osseus, An Experiment in Next Generation LVC M&S Architecture

Mark Riecken
Trideum Corporation
Orlando, Florida
mriecken@trideum.com

Derrick Franceschini
StackFrame, LLC
Sanford, Florida
derrick@stackframe.com

Scott Gallant
Effective Applications
Orlando, Florida
scott@EffectiveApplications.com

John Rutledge
Trideum Corporation
Huntsville, Alabama
jrutledge@trideum.com

Walter Barge
Director, Joint Assessment and Enabling Capability (JAEC)
OSD Force Readiness and Training (FRT)
Alexandria, Virginia
walter.s.barge.civ@mail.mil

INTRODUCTION

Periodically it is essential for a technology-supported community, like the Live, Virtual, Constructive (LVC) modeling and simulation (M&S) community, to consider whether its technology base is adequate for current needs and whether that base will sustain the community into the future. Computing and networking technologies are evolving at an astonishing rate (International Telecommunications Union, 2014)—new technologies appear on the scene and old technologies are deprecated, significantly revised or are otherwise unsustainable. Application areas that do not adapt to new technologies will face increasing financial costs and challenges in their ability to fully execute their mission. This paper provides the summary results of a small research effort, sponsored by the Defense Modeling and Simulation Coordination Office (DMSCO), to develop concepts and a prototype for the next generation LVC M&S architecture. Although this effort focused on distributed training, the results apply to many other M&S use-cases. The results of this effort indicate that there are good reasons to consider some major new ideas, especially in the area of refresh of fundamental technology, to meet increasing cost and complexity challenges. Rather than continue to adapt aging core-standards and architectures to modern web-based technologies, it may be time to consider the opposite approach: build a new core-framework and adapt legacy systems to it. Further, this effort shows that it is reasonable to consider automating some aspects of interoperability to eventually create a software "interoperability engine."

A SHORT HISTORY OF LVC ARCHITECTURE

All U.S. Armed Services, individually, and at the Joint level, enjoy significant maturity in their use of LVC or LVC-Gaming (LVC-G) distributed M&S architectures, achieved over the last 25 to 30 years with considerable investment and measurable success. The architectural underpinnings of this success may be beginning to experience pressure both in the ability to maintain pace with the rapidly changing and complex operational environment (OE) as well as newer technologies that need to be applied to maintain technical concurrency. Various sources have identified shortcomings (cf. Henniger, et al., 2008, USAF Warfare Center, 2013) but little has been done to address some of the fundamental issues.

Short History of Distributed M&S in the Broader Context of Technology

The history of distributed M&S is well known to the I/ITSEC community. Like many histories, it has well-documented dates but the broad arc describing that history unfolding is open to debate. Figure 1 characterizes the history of distributed M&S since 1990 (Rutledge, Riecken, Franceschini, & Gallant, 2015). The first phase of development of the 1990s shows the *Pre-Internet* era that saw the initial development of the DIS standard. In the

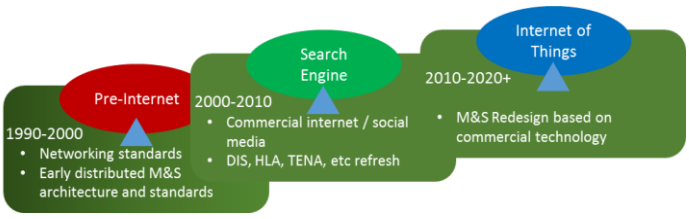

Figure 1 Concise History of Distributed M&S

second period of this history, the *Search Engine* era, commercial technology arguably outpaced the development of the DoD M&S standards, although significant improvements and upgrades to these standards occurred. In the current era, the world is experiencing a significant uptick in scale and data in internet devices and web-based technologies (Pew Research, 2014). The authors refer to this as the era of the *Internet of Things*.

Current State of LVC M&S Architecture

The current state of all LVC architectures relies on one or a combination of architecture standards and protocols, notably Distributed Interactive Simulation (DIS) (IEEE Standards Organization, n.d.), High Level Architecture (HLA) (IEEE Standards Organization, n.d.), Test and Training Enabling Architecture (TENA) (Test Resource Management Center, n.d.), and the Common Training Instrumentation Architecture (CTIA) (U.S. Army, PEO STRI, n.d.). The developers of these architectures and protocols founded them on the use of Transmission Control Protocol/Internet Protocol (TCP/IP) or similar network mechanisms, but all essentially pre-date the internet era. This is not to say that there have not been significant technology refreshes, but, arguably, they have done little that fundamentally changes their foundations to take full advantage of modern, web-based technologies. This would be only of passing historical interest if the problems with the current state were not growing (Henniger, et al., 2008). Some technologies stay usable for long periods of time with only minor upgrades. Other technology standards such as the electrical wall socket have enjoyed relatively long lifetimes. Technologies in the heavily-versioned software-world such as browsers and display technologies have arrived on the scene and nearly completely disappeared in less time than the history of distributed LVC.

APPROACH

The team used a systems engineering approach to identify the issues and decompose those issues into user stories and requirements. Given the small size of this effort, DMSCO identified distributed training as the single use case for immediate focus. Based on this use case, the team developed a prototype capability to meet its requirements and a concept of operations (CONOPS) for the vision of the objective state, i.e., the next generation LVC architecture.

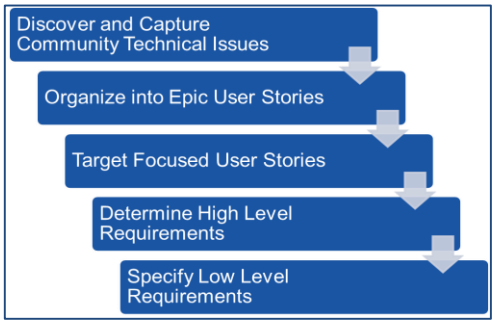

Figure 2 Systems Engineering Approach

Systems Engineering

Figure 2 illustrates our approach to developing a complete systems engineering traceability tree. The first step is to discover and capture the community technical issues. These issues come from a wide variety of sources including recent M&S architecture reviews and assessments conducted by the Services and other agencies. Their comprehensive nature certainly precludes the possibility that they are all addressed by this effort. On the other hand, these top level community technical issues form a sound baseline from which to conduct continued systematic research. The team then mapped the community technical issues to one or more "Epic User Stories." Epic user stories capture broad user objectives that still need to be further decomposed into lower level user stories, referred to here as "Focused User Stories." Focused user stories are at the level that a development team is comfortable using for development purposes.

High level Issues

The team, applying its collective experience, identified 46 high-level issues (viz.,Rutledge, Riecken, Franceschini, & Gallant, 2015) using a variety of sources from DoD and across the Services, e.g., (Allen, Lutz, & Richbourg, 2010; Allen, Gaughan, Gallant, & Schroeder; Joint Staff DJ7, 2014); USAF Warfare Center, 2013) (NATO, 2012). These high level issues formed the top of a tree structure that the team decomposed into requirements and user stories. Table 1 provides a summary of the high level issues collapsed into two summary categories. Most of the 46 issues fell into the first category: the complex, highly technical environment.

Table 1 Summary of High Level Issues

Issue Summarization	Discussion
Distributed M&S is a complex, highly technical environment	• Some M&S architectures, especially those supporting complex training environments, are themselves equally as complex and highly technical. • These environments require specialized knowledge and skills and therefore, in some cases, can limit the ability of the training audience to be flexible and adaptive in an increasingly complex Operational Environment (OE) in which training needs can change quickly. • For example, one source states that "It seems desirable to minimize if not avoid the current need for federation agreements, gateways, and recoding among users and developers often required to conduct LVC simulation activities. Current technologies (HLA, DIS, TENA, and CTIA) address the issue, but tend to particular applications, and are not inherently compatible, hence the need for agreements and fixes to make them [interoperate]."
Originally developed in the pre-internet era; not taking advantage of current technologies	• Although distributed computing was available to the architects of current and legacy M&S architectures, this work pre-dated the "modern" internet. • There are many major technologies that are now available that could support a more advanced M&S distributed architecture. • One source states that "In stark contrast [to the DoD M&S sector], the commercial sector has implemented network and browser based solutions [that] enable seamless, on the fly web-based interoperability. Military M&S needs are complex and demanding; yet, if disparate companies in the marketplace can solve the shared infrastructure, interoperability, and composability problem, then fixing defense M&S should also be doable, jointly benefitting the Training, Testing, Planning, Acquisition, and Analysis communities within DoD."

In the next step, we mapped the focused user stories into high level requirements (HLR) and finally low level requirements (LLR). The requirements are in the form of verifiable "shall" statements. The team employed the user stories and the requirements in this project to provide guidance to the prototype development team. This comprehensive systems engineering approach has the advantage that multiple research efforts can continue to benefit from the same systems engineering tree and still maintain traceability to the top level community technical issues.

Use Case – Distributed Training

This project considered a single use case, referred to as the "Develop and Manage LVC-based Training Event," shown in Figure 3. The use case involves the following four roles:

- The LVC-based Event Manager (this is like a "sim center" manager)
- The non-expert user (probably a Warfighter)
- The M&S Engineer (most like a software developer)
- The LVC Engineer (most likely a system integrator)

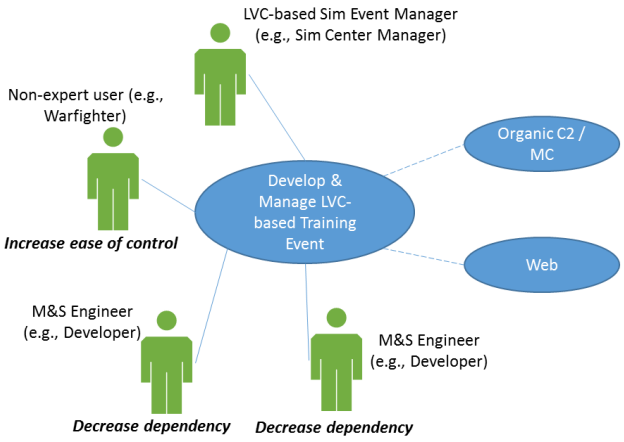

Figure 3 Distributed M&S Use Case - Training

In order to meet the needs of a more open and flexible approach to creating and managing distributed training events, the team considered the following attributes for these roles. The Warfighter needs to decrease dependency on both the engineers (developer and integrator), but increase his or her ease of control in this use case. Increasing the non-expert user's role does not mean adding additional workload or asking the non-expert user to undergo training over and above the training he or she has already taken. Rather, it means increasing the capability and the flexibility of that non-expert user to take actions more independently of the developers and integrators. The dependency for the training event on the LVC-based event manager may decrease as well since the role of brick and mortar simulation facilities may be significantly reduced given the transition of applications to cloud-based data centers (Department of Defense, Chief Information Officer, 2012) (Under Secretary of the Army, 2013).

This use case diagram also shows the incorporation of web-based access to M&S assets in keeping with obvious conclusion that this is an integral feature of modern distributed training environments. In addition, the team included the "organic" command and control (C2) or mission command (MC) devices and networks since these are essential to the "train as you fight" paradigm.

Prototype Development and Design Objectives

In order to demonstrate certain key next generation architecture concepts the team developed a prototype to demonstrate the viability of making progress of meeting some of the LLRs in the near term. This software platform is referred to as *Osseus* and is discussed in subsequent sections of this paper. Design objectives for the Osseus prototype were as follows:

Efficient Connectivity Setup. Osseus leverages the emergent WebSocket standard for web-centric bi-directional communications. WebSockets operate as TCP connections initiated over the HTTP protocol, and are lightweight and easily routable across network boundaries as well as securable via the same Secure Socket Layer (SSL) mechanisms used in HTTPS. Traditional DoD protocols require complicated mechanisms for multicast routing that are not readily supportable across network boundaries. With the Osseus prototype, users can interact with the simulation by directing their client at a uniform resource locator (URL) endpoint, similar to how users direct their web browsers to their favorite search engine. Further, the Osseus server advertises itself on the network allowing easy discovery (e.g., Zeroconf) of the server's endpoint, eliminating additional configuration challenges.

Simple Data Sharing. Osseus uses JavaScript Object Notation (JSON) as an encoding mechanism to share state. This is the encoding scheme widely used in commercial development: it offers similar benefit to Extensible Markup Language (XML) representations at a significantly reduced size. JSON data is easily consumed by web applications, and is simple to generate and consume across platforms and languages. Although a text-based format, straightforward techniques can be applied to efficiently reduce its size.

Transport Efficiency. The Osseus prototype leverages compression to reduce bandwidth. With negligible central processing unit (CPU) impacts the Osseus prototype achieves four to eight times reduction in over the wire bandwidth usage over sending uncompressed. Additionally, the Osseus prototype provides geographic filtering services to reduce unneeded simulation content sent to participants. This capability allows clients to dynamically update their filter, such as using the center point and radius of their viewport, to allow the Osseus server to eliminate network updates for objects and events outside the specified area of interest.

Data Persistence. Applications are not always connected to receive state since the beginning of an exercise, and with efforts to reduce bandwidth usage, are often only provided a filtered slice of the contents of a distribution. Persistence provides a mechanism for late-joining nodes to get caught up with the current state of the distribution without relying

on network intensive approaches. Osseus includes both a NoSQL database solution and a Time Series Database (TSDB) solution for persistence. This allows dynamic flexibility in design of data model schema, promoting adaptability between disparate data model definitions.

Behavior Authoring and Editing by Non-Expert Users. Frequently users discover gaps in capabilities within a collection of applications that must be addressed to support a training objective. Historically, this requires the execution of software development to fill that gap. Osseus includes the ability for non-expert users to create and modify behaviors, which reduces the expense of traditional software development cycles to fill capability gaps. Leveraging an Open Source toolkit, the Osseus Prototype enables non-expert users the ability to graphically design a behavior from pre-built components. This simplistic interface generates JavaScript representations of the resultant behavior, which can be persisted and dynamically executed within the Osseus platform, affecting the Osseus data model.

Support for Finer Granularity Models. The Osseus prototype support incorporation of services or models as opposed to full applications. This makes integration of components, for example a component providing consistent damage adjudication services for the exercise, significantly easier than previous efforts. Many models within DoD aren't written to be used within a System of Systems simulation environment: just grabbing the model and wrapping it in Osseus is easier and more flexible than requiring the model to be incorporated into an existing simulation. Within Osseus, this can be done by creating a service. In the simplest form, this could be made a web service.

Performance Measurement. The ability to capture and visualize performance characteristics of a running system is critical to understanding and tuning the behavior of a distribution. The Osseus prototype includes built-in metrics and visualization tools to understand tune the behavior of the Osseus Platform. This includes memory and CPU usage of the Osseus server, number of connected clients, and distribution bandwidth usage. This data is dynamically graphed against time to provide a dashboard view of the performance of Osseus. This extensible capability leverages Open Source tools to persist and visualize performance characteristics of the platform.

NEXT GENERATION ARCHITECTURE CONOPS

The next generation architecture requires a concept of operations (CONOPS) for the distributed training use case. Terminology associated with and characteristics for that CONOPS follow.

Some Next Generation Architecture Concepts

As a first step in providing a more open and flexible interoperability, the team introduced the concept of an "alliance." An alliance is a collection of distributed applications, services, and models brought together for a training or analytical purpose. Alliances are intentionally less structured than HLA federates: data models are discovered dynamically rather than worked out through consortium *a priori*. Alliances are made up of affiliates which are applications, services, tools, and/or models. Affiliates are less rigidly bound to a data model than HLA federates are: there is no *a priori* Federation Object Model (FOM) or Runtime Infrastructure (RTI) Initialization Data (RID) file to conform to with this mechanism. A software platform, Osseus, is the set of services, tools, data, and models that support affiliates and alliances sharing information. An Osseus Object is a defined instance of a collection of attributes. Objects are defined in the Osseus Data Model (DM). There can be multiple unique instances of objects in an Osseus Alliance. Objects are persisted within the Osseus server.

Objective State

Figure 4Figure 4 illustrates a view of the CONOPS for the objective state of the next generation architecture. Starting from the left, the vertical box labeled Distributed Joint Training Needs indicates the body of information that the "non-expert user (trainer)" uses to develop a distributed training event. The non-expert user may be a military unit commander or leader with considerable subject matter knowledge of tactics, techniques, and procedures (TTP) and doctrine yet may not have a comparable level of expertise with distributed M&S LVC environments. This user interacts with an Osseus server or servers to find LVC configurations that most closely meet the needs of the training event. Osseus is continually updating its information about available LVC configurations, applications, and models. Based on technologies such as Puppet, Osseus would be able to recommend an LVC configuration and then, after necessary adjustments are made by the user and an expert (human) assistant, Osseus can execute the event using the identified resources. Even in a complete realization of a next generation distributed training architecture, it is likely

that the non-expert user will need at least the standby assistance of an M&S expert. This is accounted for by the presence of the second user (Expert Assistant) in the graphic.

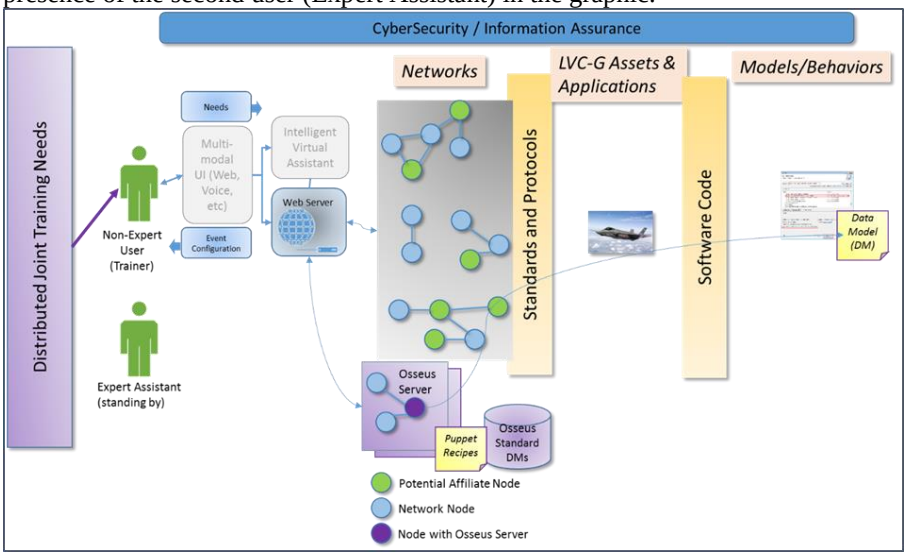

Figure 4 A Next Generation Architecture CONOPS

This objective CONOPS addresses the key desired characteristics of the next generation architecture including 1) more open and flexible interoperability between disparate systems; 2) the ability for relatively untrained users to fill in functionality gaps between available systems by dynamically injecting behavioral changes into the LVC environment; 3) the ability to connect services that are smaller in granularity than an application to increase the capability of the environment; 4) a more accessible means of putting together distributed training capabilities to an educated, but non-specialist trainer; 5) data filtering to optimize or reduce data transmission over the network; and 6) centralized data management to facilitate tools such as visualization, data collection, and analysis.

A PROTOTYPE PLATFORM FOR THE NEXT GENERATION: OSSEUS

Figure 5 shows a context diagram for the prototype Osseus which represents a first step in implementing the next generation vision discussed abobe. The Osseus server provides services to persist current simulation state, events, and behaviors. This server also provides services to reflect object and attribute updates out to registered clients, provides mechanisms to filter data to clients, and provides a platform to execute behaviors to change simulation state. As part of this prototype effort, the team constructed a two-dimensional thin client view leveraging Open Source tools and data for the map framework, as well as Open Source tools for the Icon Symbology service. Additionally, the team constructed an application with a three dimensional view leveraging the commercial Unity Game engine to ensure the data transmitted through Osseus supports virtual use cases. Figure 6Figure 6 is a more detailed view of the Osseus platform architecture.

2D viewer. As shown in the figure, there is a web-based 2D view allowing Osseus users to visualize the geospatial representation of Osseus data set. This view is created in a web browser by combining the services of open source data for the map from the OpenStreetMap project, with icon Symbology from the Open Source Mission Command Mil-Sym-Service web application. These pieces are combined with another JavaScript framework called OpenLayers, to present a functional 2D map capable of panning and zooming. For this effort, a JavaScript based Osseus interface was created to subscribe to Osseus objects and events,

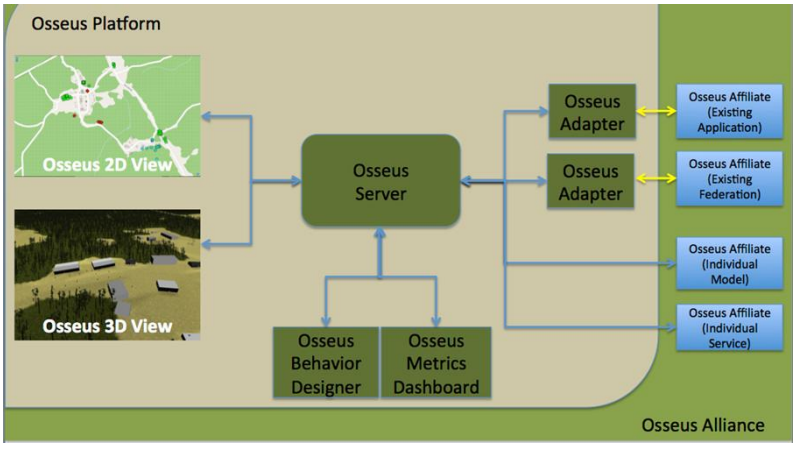

Figure 5 Osseus Context Diagram

publish the viewport of the map, and render entity icons and detonations.

3D viewer. The team constructed a 3D virtual/game view built using the Unity game engine; the StackFrame Magrathea terrain generation application was used to create the 3D terrain scene. This combines open data for elevation and features into a correlated Objective Terrain Format (OTF) format and formats consumable by 3D engines such as Unity. 3D actor models and animations were incorporated from Unity's asset store. Simulation data into the 3D view is provided through a C# interface listening to Osseus objects and events and translating those into scene updates. The 3D view also publishes its camera location out through Osseus so other affiliates can view the camera location. Additionally, the 3D view registers for a geospatial-filtered view driven by the published camera location and a view radius. The 3D view allows visualizations of aspects of the data problem pertinent to a Virtual/Game domain. Of note, neither the 2D view framework nor the game engine framework had native support for an existing DoD middleware or distributed protocol.

Test harness. To stimulate the prototype (i.e., provide a sample Affiliate) the team leveraged the constructive OneSAF application. To adapt OneSAF content into Osseus, our team built a runtime deployable web component that could be dynamically registered with OneSAF for execution, without modifying any OneSAF code. This data adapter translates the internal OneSAF data model into the Osseus data model client-side, packages it the JSON format, and sends it via WebSocket to the Osseus server for distribution.

Services. Osseus services provide communication with systems external to the Osseus server itself using an Osseus protocol. The Osseus server also supports the offloading of services from the Osseus server itself. Examples of Osseus services include: Remote behavior engine service; Dynamic terrain server; Damage effects; and Heart beating. Osseus has *internal services* to facilitate concepts such as data model adaptation to other middleware products. Data adapters are tightly coupled to the Osseus data model and provide bi-directional transformations between Osseus native and other formats. These provide efficient, prebuilt mediation services. Other services can also be provided such as behavior engines for behavior execution, time and state management services for connected simulations and clients. An Osseus *service proxy* connects Osseus clients to a remote service oriented architecture (SOA) component. An example of a service proxy would be a service, which translates Osseus entity states into C2 Location reports for a tactical network. An Osseus *service relay* connects Osseus Clients to Osseus Services running on networks, which may or may not be, accessible by the Osseus Client. The relay is similar to a network router in that a given client does not know nor need to know how to deliver data directly to the destination. Osseus provides a mechanism for services to *advertise* their availability in a unified way. *Advertisement* allows clients to discover capabilities or resources for easier access. This is used for non-Osseus services to be consumed by other Osseus services or clients. An example of an external service is a REST interface to geospatial database to enable requests for height of terrain at a location or line of sight (LOS) between two end points. Another example of an external service is the URL to a browser-based Osseus client.

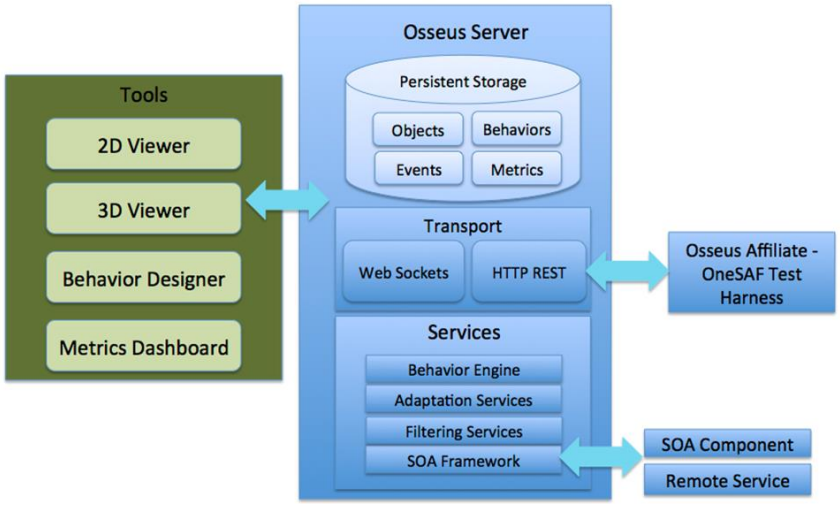

Figure 6 Osseus Platform - Detail

Persistent storage. The Osseus prototype provides persistent storage of behaviors, events, and objects. Data are stored in Osseus native formats. NoSQL database technology will be utilized to repose the data model. Specifically, the team utilized a document oriented NoSQL database to allow flexibility in the data model representation without the necessity of updating schema definitions.

Data Model. Osseus maintains an internal data model describing simulation state and events. The data model is client-extensible. Clients can register additional object or attributes on existing objects. This data model is expressed in JSON representation, providing a lighter-weight representation than equivalent XML. The data model includes object representations for entities, as well as battlespace events such as weapon fire and detonations. It also includes simulation control state representation, to describe if the simulation is running or paused. Objects like entities include geospatial representation as well identifiers for type.

Bindings. Osseus will have language-specific data bindings so that clients can access Osseus data from a variety of development platforms. These bindings enable easy client access to the data model, regardless of the client programming language. The data bindings exist for since it will be the native data model format. A capability exists to generate JavaScript bindings directly from the Java data model on demand. Currently, a static version of the data-bindings exist for C# with plans to automatically generate these in the future.

Transport. Osseus will take advantage of modern web transports to help lower the barrier to entry for clients. One of the reasons for selecting web-based transports is to allow for browser-based tools to access all data and services within Osseus. Further, the transport leverages Hyper Text Transmission Protocol (HTTP), to simplify routing through Wide Area Networks. Osseus supports Representational State Transfer (REST) over HTTP connections for interacting with the data model and Osseus services. HTTP REST provides a client-friendly method to do create, read, update and delete (CRUD) operations from a variety of platforms and languages. Osseus supports stateful connections between client and server using WebSockets. WebSockets provide bi-directional from servers to clients that enable easy routing through the Internet. WebSocket clients use a lightweight protocol for distinguishing between object, event, and service-centric messages.

Behavior authoring. The Osseus prototype provides a mechanism for non-expert users to alter the behavior of objects within an Osseus Alliance. Building behaviors or models is a significant cost driver for resolving capability gaps in a simulation system. This typically requires skilled software engineers to translate subject matter expertise into functioning software, an expensive effort in both time and money. The Osseus prototype provides a graphical tool for non-expert users to build complex behaviors from primitive logic and behavior building blocks. This is consistent with modern game engines mechanisms that enable developers to graphically script activities for any object in a scene. The team used a visual building block approach leveraging an open source implementation of a visual programming tool originally designed to teach kindergartners to program. Users drag and drop building blocks together to create new behaviors. Interfaces between components are strictly defined allowing the framework to provide visual and aural queues when a user can connect two components. The resultant behaviors are compiled into JavaScript executable behavior representations Osseus provides a visual tool for building complex behaviors from primitive logic and behavior building blocks. The resultant behaviors are code-generated into JavaScript executable behavior representations.

Built-In Metrics. The Osseus Platform is predicated on the idea that many real-time adjustments can be made to optimize the performance of an affiliation, and that instrumentation and visualization are critical to understanding and predicting performance of the distribution. To this end, metrics collection and visualization are built in to the platform for the purpose of measuring various performance aspects of an affiliation. The Osseus platform routinely writes out data to a Time Series Database (TSDB), and an Open Source thin client library for graphing performance metrics can view current metrics or move back to other collected points in time to view. The graphical dashboard view for metrics is configurable so that different aspects of collected metrics can be shown. For bandwidth usage comparison as part of this metrics collection, Osseus also logged the outgoing DIS traffic performance for the same exercise. While DIS consumed more than 140Kbytes/sec in network bandwidth, Osseus consumed less than 10Kbytes/sec over-over-the-wire bandwidth. For comparison sake, the team examined sampled three scenarios of different sizes to compare over-the-wire bandwidth usage and measured both native Osseus bandwidth and DIS bandwidth. See Table 2 for details.

Table 2 Osseus Prototype Measured Bandwidth

Entity Count	DIS average bandwidth	Osseus average bandwidth
100	40kb/s	8kb/s
1000	400kb/s	20kb/s

Entity Count	DIS average bandwidth	Osseus average bandwidth
3000	1.2mb/s	40kb/s

The difference in bandwidth used is substantial and grows as the exercise density increases. This difference is due to the difference in approaches: DIS heartbeats objects while Osseus does not; DIS send full object updates while Osseus prefers to send only object property changes, and Osseus uses compression on the wire where DIS does not. The team has also measured Osseus performance at significantly higher exercise densities, including 100,000, 200,000, 500,000, and 1,000,000 entity densities, and are confident that Osseus scales well in higher densities.

RESULTS: CHARACTERISTICS OF OSSEUS AS A NEXT GENERATION ARCHITECTURE SOLUTION

This effort has shown that through the application of modern web-based technologies, many desirable characteristics of a next generation distributed M&S architecture can be achieved. Osseus is Government Purpose Rights (GPR) software and is at the core of our prototype for this next generation architecture. The ultimate goal of this effort is to establish a free and open source (FOSS) software platform. Table 3 summarizes the Osseus approach in addressing the six characteristics of a next generation architecture.

Table 3 Summary of Osseus as Next Generation Architecture Approach

	CHARACTERISTIC OF NEXT GENERATION ARCHITECTURE	OSSEUS APPROACH
1	Open and flexible interoperability between disparate systems	The Osseus platform uses a variety of mechanisms including its own data model, as well as adaptation services to lower the barriers to interoperability. Although OneSAF and an open source virtual (3D) affiliates were used in our prototype experiment, any current constructive or virtual application could participate in an Osseus affiliation. This is also a significant first step towards automating aspects of interoperability creating an "interoperability engine."
2	Ability for relatively untrained (non-expert) users to fill in functionality gaps	Osseus content authoring, based on an open source capability and JavaScript code generation, facilitates ease of access to this capability.
3	Ability to connect services that are smaller in granularity than an application	The Osseus platform enables the ability to use models within an application without having to deploy the entire application.
4	Accessible means of composing distributed training capabilities for an educated, but non-specialist trainer	The Osseus platform prototype provides capabilities and tools that can support this capability, but further work is required to develop suitable UIs.
5	Data filtering to optimize or reduce data transmission over the network	The Osseus platform provides filtering services; the prototype demonstrated that significant bandwidth traffic reductions are possible.
6	Centralized data management to facilitate tools such as visualization, data collection, and analysis	The Osseus platform includes 2D and 3D visualization as well as built-in metrics display capability.

SUMMARY AND LESSONS LEARNED

This paper has introduced the concept of the Osseus Platform to address some of the high level community issues assembled during a DMSCO-sponsored research effort. This effort developed a set of systems engineering products including requirements and user stories. This effort also produced a working GPR prototype with many features of a next generation distributed M&S architecture that would be open source and have advanced features such as an "interoperability engine" that would greatly simplify user interaction with LVC architectures. The results showed that it is possible to start from an updated technology base and rapidly develop a viable alternative to current and legacy M&S architectures. This result is of value to the Training community as well as other M&S communities of practice.

The Osseus approach starts with a core of modern, web-based technology and facilitates legacy adaption - instead of using legacy core technology and adapting to modern technology.

Our lessons learned can be summarized as the following:

- An adaption platform can ease interoperability barriers
- There is a need for technology refresh to keep pace with commercial advancements
- Behavior (and content) authoring can be simplified for non-expert users
- Measures can and should be "built-in" to the architecture
- Focusing on automating interoperability could provide significant dividends

The Osseus prototype is a step towards a full-fledged Osseus Platform that is free and open source software for the M&S community. In the short term, there is potential to optimize interoperability and some distributed M&S configurations with tools and techniques developed in this phase of the project. In the long term: development of an Osseus platform with a robust *automated interoperability* capability may prove essential to maintain pace with both technology and the complexity of the Operational Environment.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the support of Mr. Ralph Gibson; the insights and contributions of Mr. Gene McCulley and Mr. Jared Davies; and the careful reading and editing of Ms. Lisa Bair.

REFERENCES

Allen, G. W., Lutz, R., & Richbourg, R. (2010). Live, Virtual, Constructive Architecture Roadmap Implementation and Net-Centric Environment Implications. *31*.

Allen, G., Gaughan, C., Gallant, S., & Schroeder, L. (n.d.). Vision of a Composable Architecture Simulation Environment (CASE).

Department of Defense. (2015, January 16). *Virtual World Framework*. Retrieved from Virtual World Framework: https://virtual.wf/

Department of Defense, Chief Information Officer. (2012). *Cloud Computing Strategy.* DoD.

Gaughan, C., Metevier, C. J., Fogus, M., McDonnell, J. S., & Gallant, S. (n.d.). Executable Scenario Definition Using Datalog to Describe Simulation Capabilities.

Henniger, A. E., Cutts, D., Loper, M., Lutz, R., Richbourg, R., Saunders, R., & Swanson, S. (2008). *Live Virtual Constructive Architecture Roadmap (LVCAR) Final Report.* Institute for Defense Analyses.

IEEE Standards Organization. (n.d.). *1278.1-2012*. Retrieved June 5, 2015, from IEEE Standards: https://standards.ieee.org/findstds/standard/1278.1-2012.html

IEEE Standards Organization. (n.d.). *1516-2010*. Retrieved June 5, 2015, from IEEE Standards: https://standards.ieee.org/findstds/standard/1516-2010.html

International Telecommunications Union. (2014). *Measuring the Information Society.* ITU.

Joint Staff DJ7. (2014). *DRAFT Joint Training Environmnent Information Systems Initial Capabilities Document (IS-ICD).*

Martin E. Dempsey, G. U. (2012). *MIssion Command White Paper.*

Martinez-Salio, J.-R., Lopez-Rodriguez, J.-M., Gregory, D., & Corsaro, A. (n.d.). A Comparison of simulation and Operational Architectures.

NATO. (2012). *NATO Modelling and Simulation Master Plan.* NATO.

Pew Research. (2014, March 14). *Chart of the week: the ever-accelerating pace of technology adoption*. (D. Desilver, Editor) Retrieved June 11, 2015, from www.pewresearch.org: http://www.pewresearch.org/fact-tank/2014/03/14/chart-of-the-week-the-ever-accelerating-rate-of-technology-adoption/

Rutledge, J., Riecken, M., Franceschini, D., & Gallant, S. (2015). *Distributed Architecture Technology Review.* Submitted to Defense Modeling and Simulation Coordination Office (DMSCO).

Test Resource Management Center. (n.d.). *TENA Software Development Activity*. Retrieved June 5, 2015, from www.tena-sda.org: https://www.tena-sda.org/display/intro/Home

U.S. Army, PEO STRI. (n.d.). *Common Training Instrumentation Architecture*. Retrieved June 5, 2015, from PEO STRI Products: http://www.peostri.army.mil/PRODUCTS/CTIA/

Under Secretary of the Army. (2013). *Migration of Army Enterprise Systems/Applications to Core Data Centers.* DoD.

USAF Warfare Center. (2013). *Live, Virtual, and Constructive (LVC) Summer Study (Draft).* Nellis AFB, Nevada.