Leveraging MSaaS Concepts to Enable Mission Environments: Lessons Learned

Jay Freeman, Evan Oster, Stanislav Ivchenko
CAE USA
Orlando, FL

Andreas Krupp, Máté Koch
CAE Germany
Stolberg, Germany

Erik Bernheim
CAE USA
Tampa, FL

ABSTRACT

The 2022 National Defense Strategy (NDS) from the United States, and other similar documents from Allied and NATO member nations, describe a global condition of near peer threats: a 'decisive decade' where integrated deterrence is a core element. Modelling and Simulation (M&S) is a key technology for integrated deterrence, which requires collaboration with Allied and NATO partners in large scale and multinational exercises. To address the challenge of creating a heterogeneous simulation environment for integrated deterrence, we implemented the Modelling and Simulation as a Service (MSaaS) Technical Reference Architecture established by NATO MSG-136.

MSG-136 provides technical guidelines, recommended standards, and architectural building blocks for constructing M&S capabilities. The MSaaS concepts mirror those used in cloud computing and service-oriented architectures. Our approach utilizes a dual Consumer Layer (web-based thin client and immersive, native client) that is agnostic to the underlying simulations within the Service Layer. This Service Layer integrates multiple third-party Computer-Generated Forces (CGFs). A common data layer stores terrain, order of battle, and scenario information. The Operational System Layer features CGF-agnostic behavior of entities and units, an analytics-ready data strategy, and cloud deployment. This architecture enables rapid application creation for various Live, Virtual, and Constructive (LVC) use cases by implementing services for discovery and composability. While MSaaS has been applied in various exercises, there is a lack of in-depth studies on its application to large-scale LVC exercises and training events. This article provides a novel path forward as the training and M&S community works to expand the application of MSaaS and facilitate future joint and coalition interoperability.

ABOUT THE AUTHORS

Jay Freeman works for CAE USA as a Synthetic Environment Technical Fellow and oversees several projects at CAE. Mr. Freeman previously served as the System and Software Architect for multiple DoD programs. He attended Hobart College for undergraduate studies and the University of Alabama in Huntsville for graduate studies. Email: Jay.Freeman@caemilusa.com

Evan Oster works for CAE USA as a Product Manager for modeling and simulation products. His education includes a BS in Human Development at Virginia Tech, MS in Curriculum and Instruction at Radford University, and MA in Instructional Design and Technology at Virginia Tech. Email: van.oster@caemilusa.com

Máté Koch works for CAE Germany as Innovation and Technology Lead, managing international teams involved in innovation, R&D and product development. He holds an MSc in Computer Science and Physics from Eötvös Loránd University, Budapest and a MIT xPRO certificate for Technology and Innovation Acceleration. Email: mate.koch@cae.com

Erik Bernheim works for CAE USA as a Lead Software Engineer primarily in R&D. He holds a B.S. and a M.S. in Business Analytics and Information Systems from the University of South Florida. Email: Erik.Bernheim@caemilusa.com

Andreas Krupp works for CAE Germany as Group Leader of the Synthetic Environment group, leading an international team of software developers and architects as well as supporting business development in technical matters. He holds an MSc in Physics and Computer Engineering from Heidelberg University. Email: andreas.krupp@cae.com

Stanislav Ivchenko works for CAE USA as a Technical Professional in research and development. He holds a B.S. and M.S. in Computer Science from the University of Central Florida. Email: stanislav.ivchenko@caemilusa.com

Leveraging MSaaS Concepts to Enable Mission Environments: Lessons Learned

Jay Freeman, Evan Oster, Stanislav Ivchenko
CAE USA
Orlando, FL

Andreas Krupp, Máté Koch
CAE Germany
Stolberg, Germany

Erik Bernheim
CAE USA
Tampa, FL

INTRODUCTION

The ability for military forces to achieve mission-readiness in complex and large-scale scenarios increasingly relies on digitally recreating mission environments (Perey et al., 2022). With modeling and simulation tools and scalable design, complex scenarios can be realistically simulated across military domains (e.g., land, air, maritime, cyber, space) and non-military domains (e.g., economic, technological, information). These mission environments serve a range of purposes from doctrine definition to training and wargaming. Digital mission environments have been exploited for decades, but the recent re-emergence of near-peer threats necessitates a shift in focus to increase interoperability for integrated deterrence (Joint Chiefs of Staff, 2015).

The 2022 National Defense Strategy (NDS) identifies top-level defense priorities of the United States. One of these top-level priorities is “integrated deterrence” which entails using every tool at the Department’s disposal, in close collaboration with our counterparts across the U.S. Government and with Allies and partners. To achieve integrated deterrence within the Modeling and Simulation (M&S) community, the constraints of stove pipe training systems must be eliminated to ease the integration of the Ally and partners’ disparate systems.

Integrated deterrence extends beyond traditional conflict to include gray zone activities from near peer threats at a campaign level, Gray Zone activities are generally defined as behaviors and/or actions potentially leading to, but below the threshold of armed conflict executed across actors’ instruments of national power present significant national security and global stability challenges (Ducharme et al., 2023). Within the mission environment this translates to integrated large-scale conflicts and gray zone activities with near peer threats across millions of live, virtual, and constructive (LVC) entities and several domains (e.g., military and non-military). Achieving LVC exercises at this scale with realism presents several inherent challenges given the limitations of the today’s Computer Generated Forces (CGFs).

The limitations of the current generation of CGFs inhibit accomplishing the priorities described by the NDS in three primary ways. (1) Many CGFs are designed and optimized for a specifical domain (e.g., air, land, maritime, space, cyber); (2) Many CGFs are designed and optimized for a mission set (e.g., intelligence, surveillance, reconnaissance); and (3) CGF with high-fidelity models are often restricted for global use because of security and intellectual property (IP) considerations. The implications of these limitations often yield an LVC exercise with (1) only a kinetic focus, no gray zone activities; (2) little to no technology interoperability, scaling, and federation required for all domain conflict effects; and (3) unrealistic entity behavior with limited capabilities potentially resulting in poor doctrine, negative training, and ineffective mission Courses of Action (COAs). It is unrealistic at present time to expect any single CGF system to serve the US and Allies, support millions of LVC entities, and cover the entire spectrum of military and non-military capabilities. Even if this were possible, it may include undesirable side-effects such as vendor-locking the government into single solution, reducing interoperability with other capabilities, raising concerns about data ownership, and inhibiting innovation. Hence the only viable solution is the interoperability of multiple disparate systems.

To achieve this interoperability and accomplish the needs outlined above in the NDS a Modular Open Systems Architecture (MOSA) and cloud agnostic approach is required to deliver the scale and complexity of the mission environment (Defense Standardization Program Office, n.d.). MOSA design includes highly cohesive, loosely coupled, and severable modules which can be acquired from independent vendors and enable the system to evolve with future technologies and capabilities. Delivering MOSA solutions in modern cloud environments includes employing containerization, composability concepts, cross-platform approaches, and API/Software Development

Kits (SDK) in multiple forms, enabling technology to scale, compose, and cluster functionality as needed on a range of infrastructure. A MOSA approach helps avoid architectural entropy and reduces engineering costs to link new capabilities. Creating an API/SDK approach to introduce new capabilities will reduce architectural decay and the effort to interface new capabilities to perform various data interactions and processing steps within workflows and pipelines.

By comparison, Modeling and Simulation as a Service (MSaaS) takes the MOSA approach and further tailors it with a beneficial design pattern unique to M&S to enable rapid application and creation at scale. Since traditional application development is a time consuming and cumbersome process due to the lack of consistently used standards, this facilitates development and interoperability of offerings for expansive LVC uses cases to meet the needs defined in the NDS. The following sections describe MSaaS, our implementation of MSaaS for large scale exercises, challenges encountered, and benefits observed.

MSAAS TECHNICAL REFERENCE ARCHITECTURE (MSG-136)

In May of 2019 NATO MSG-136 published a Reference Architecture (RA) for a concept called Modeling and Simulation as a Service (MSaaS). This initiative aimed to develop a cloud-based system, the Allied Framework for MSaaS, that would provide on-demand access to simulated training environments. MSG-136 argues MSaaS could significantly improve accessibility and collaboration for NATO forces. Their experiments validated this potential, prompting recommendations for further development and implementation by NATO and its Allies. The final product, the Allied Framework for MSaaS, includes guidelines, technical specifications, and governance policies to guide responsible use of this innovative training approach to composable simulations. Figure 1 illustrates the MSaaS concept.

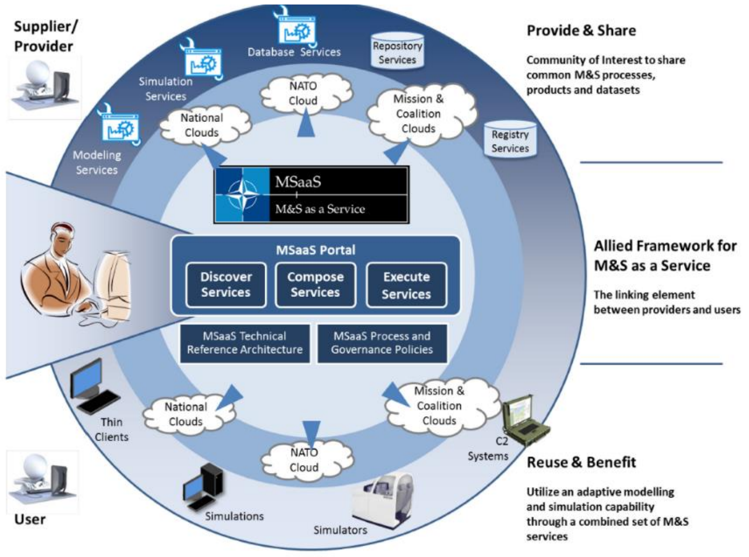

Supplier/ Provider
Database Services
Repository Services
Simulation Services
Modeling Services
NATO Cloud
National Clouds
Mission & Coalition Clouds
Registry Services
Provide & Share
Community of Interest to share common M&S processes, products and datasets
MSaaS
M&S as a Service
MSaaS Portal
Discover Services
Compose Services
Execute Services
MSaaS Technical Reference Architecture
MSaaS Process and Governance Policies
Allied Framework for M&S as a Service
The linking element between providers and users
Thin Clients
C2 Systems
Reuse & Benefit
Utilize an adaptive modelling and simulation capability through a combined set of M&S services
User
Simulations
Simulators

Figure 1: MSaaS Concept

The NATO MSaaS RA provides a blueprint for developing and using the Allied Framework for MSaaS capabilities. The MSaaS RA is a layered architecture consisting of 1) a static structure and a dynamic library of relevant Architectural Building Blocks (ABBs) and 2) Architecture Patterns (APs). MSaaS enables simulations to adhere to an enterprise-level approach for software architectures, enabling seamless integration of new technology based on well-defined interfaces and synergistic approaches to discover, compose, execute, and manage M&S services via Service Level Agreements (SLA). This provides the flexibility and agility needed to easily adapt systems as new threats and challenges arise in the global political environment, or to incorporate new technologies.

The Open Group SOA Reference Architecture (TOGSOARA) provides structure and generic ABB types relevant to any Service-Oriented Architecture (SOA). MSaaS RA outlines nine layers: 1) Operational Systems, 2) Service Components, 3) Services, 4) Business Process, 5) Consumer, 6) Integration, 7) Quality of Service, 8) Information and 9) Governance layers. MSaaS RA inherits structure from TOGSOARA, including the architecture layers, ABB types, and architecture pattern APs. It then populates the structure using ABBs belonging to the NATO C3 Taxonomy of Consultation, Command & Control capabilities.

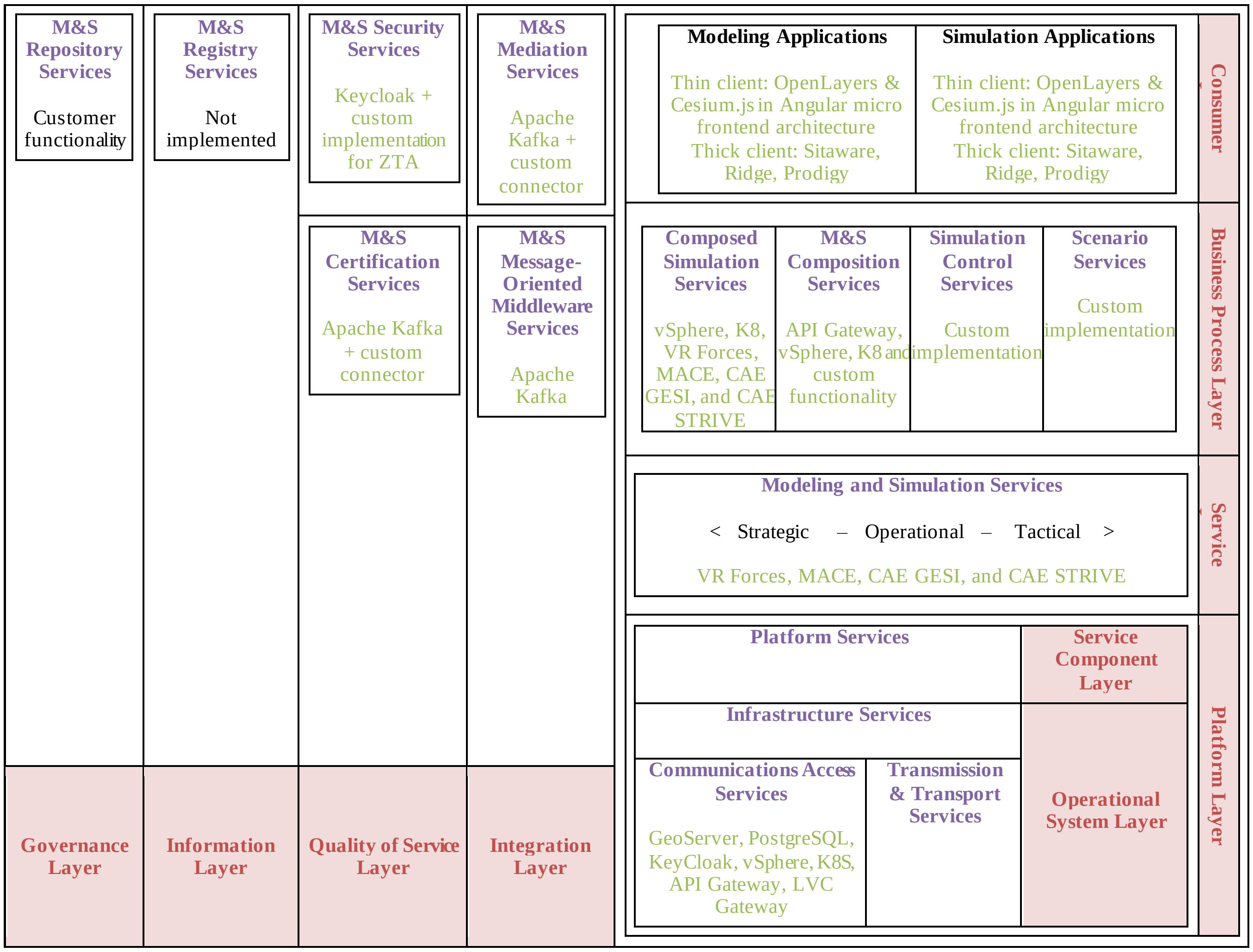
Table 1 lists the key technologies integrated into the author's prototype MSaaS implementation (for the paper, we call this "The Prototype"). The Prototype utilizes emerging technology trends and work within the NATO Industrial Advisory Group (NIAG), and standards bodies such as SISO. Several factors were used to determine which technologies to integrate (1) sustainability with respect to licensing considerations, technology and export constraints, and ability to modify code; (2) best of breed 3rd party simulation services to yield maximum utility to our implementation; and (3) MOSA implementation where technologies were required to satisfy the MOSA criteria of being modular, severable, and compatible between a variety of vendors to provide a level of ‘component supply chain resilience.

Table 1: Key ABBs and technologies integrated into The Prototype based on MSaaS guidance

MSaaS Architecture Building Blocks (ABBs)	Description	Technologies Employed
Composed Simulation Services	Totality of Simulation Services	vSphere, K8, VR Forces, MACE, CAE GESI, and CAE STRIVE
Core, Communication and SOA Services	Range of functionality enable basic system support	GeoServer, PostgreSQL, KeyCloak, vSphere, K8S, API Gateway, LVC Gateway
M&S Certification Services	Verify compliance to interoperability standards	Apache Kafka + custom connector
M&S Composition Services	Compose and execute simulation services	API Gateway, vSphere, K8 and custom functionality
M&S Mediation Services	Transform/Translate data	Apache Kafka + custom connector
M&S Message-Oriented Middleware (MOM) Services	Efficient and time-coherent exchange of data	Apache Kafka
M&S Registry Services	Metadata for simulation resources	Not implemented yet
M&S Repository Services	Store, retrieve and manage simulation resources	Federated repositories and customer functionality
M&S Security Services	Enforcement of security	Keycloak + custom implementation for ZTA
Modeling Applications	Front-end functionality for accessing the Modeling Services	Thin client: OpenLayers & Cesium.js in Angular micro frontend architecture Thick client: SitaWare, Ridge, Prodigy
Modeling Services	Suite of tools needed to construct a Simulation Service	Not implemented yet
Scenario Services	Creation & management of scenarios	Custom C2SIM implementation
Simulation Applications	Front-end functionality enabling users to interact with Simulation Services and Composed Simulation Services	Thin client: OpenLayers & Cesium.js in Angular micro frontend architecture Thick client: SitaWare, Ridge, Prodigy
Simulation Control Services	Logging and input for simulation execution	Custom implementation
Simulation Services	Individual simulation engines	VR Forces, MACE, CAE GESI, and CAE STRIVE

An interesting element of MSaaS is the lack of specifics regarding Core, Communication, and SOA Services. The rationale is these services are considered non-M&S-specific capabilities; however, these are keys to success as they are crucial elements of interoperability and scale. The Prototype is aimed to maximize scaling and interoperability in an agonistic manner, therefore exclusively open-source capabilities were selected for the Core, Communication, and SOA Services.

IMPLEMENTATION OF MSaaS CONCEPT FOR LARGE SCALE EXERCISES


M&S Repository Services
Customer functionality
M&S Registry Services
Not implemented
M&S Security Services
Keycloak + custom implementation for ZTA
M&S Mediation Services
Apache Kafka + custom connector
M&S Certification Services
Apache Kafka + custom connector
M&S Message-Oriented Middleware Services
Apache Kafka
Governance Layer
Information Layer
Quality of Service Layer
Integration Layer
Modeling Applications
Thin client: OpenLayers & Cesium.js in Angular micro frontend architecture
Thick client: Sitaware, Ridge, Prodigy
Simulation Applications
Thin client: OpenLayers & Cesium.js in Angular micro frontend architecture
Thick client: Sitaware, Ridge, Prodigy
Consumer
Composed Simulation Services
vSphere, K8, VR Forces, MACE, CAE GESI, and CAE STRIVE
M&S Composition Services
API Gateway, vSphere, K8 and custom functionality
Simulation Control Services
Custom implementation
Scenario Services
Custom implementation
Business Process Layer
Modeling and Simulation Services
< Strategic – Operational – Tactical >
VR Forces, MACE, CAE GESI, and CAE STRIVE
Service
Platform Services
Service Component Layer
Infrastructure Services
Communications Access Services
GeoServer, PostgreSQL, KeyCloak, vSphere, K8S, API Gateway, LVC Gateway
Transmission & Transport Services
Operational System Layer
Platform Layer

Legend: Services, Layers, Technologies or products

Figure 2. The Prototype implementation of the MSaaS Concept

The following section walks through the implementation of the components displayed in the figure above. A heterogenous simulation environment was employed. To validate the hypothesis of MSaaS being an enabling concept for large-scale LVC, open-source services, open standards, and 3rd party technologies were leveraged to the maximum extent possible to prevent becoming locked into specific vendors or technologies.

For the 3rd party CGFs, the Modern Air Combat Environment (MACE) and VR-Forces were integrated. MACE creates realistic combat scenarios with physics-based movements and interactions for electronic warfare, air, land, and surface training, particularly for Joint Terminal Attack Controllers (JTACs). VR-Forces creates realistic land to space terrain using real-world data to run simulations in real-time or faster. VR-Forces can model vehicles, weapons, weather, and even living things and simulate both individual units and large groups of units using the same system.

We integrated two internal products to CAE. CAE GESI is a CGF for commanders and their staff. It creates realistic simulations for complex scenarios in the land domain, ranging from company-sized groups to entire brigades. GESI offers two main training modes, one for command posts to make decisions within a simulated environment and another for classrooms to learn and experiment with tactics. CAE STRIVE is a CGF focused on the air domain. It creates realistic simulations of enemies, allies, land vehicles, aircraft, and even civilians within virtual worlds that feature accurate 3D terrain, weather effects, and sensor simulations.

The interoperability of MACE, VR-Forces, GESI and STRIVE presented a unique combination of CGFs to exercise the MSaaS concept and provided three lessons learned. First, these systems have different levels of exportability with respect to ITAR, mirroring the NATO use case. This requires the implementation to contain a mechanism to restrict the access to containers, virtual machines, and entity traffic. Second, each of these CGFs use a different set of terrain and scenario language. This requires a common database to achieve scenario and terrain interoperability (described below in the common data section). Third, each of these simulations have different concepts to entities and aggregates. This causes challenges with state management of individual entities which were deemed to be outside the scope of this implementation.

Given the combination of ITAR and non-ITAR Simulation Services, we chose to apply a Zero Trust Architecture (ZTA) pattern to enhance the security posture of the system. The DoD formalized its ZTA strategy in 2022, establishing it as a mandatory architectural framework. ZTA operates on the principle of "never trust, always verify," continuously authenticating every user and Simulation Service attempting to access resources, irrespective of their location or type. This approach minimizes access privileges and continuously monitors activity to prevent unauthorized lateral movement within the system in the event of a breach. By employing ZTA, we achieve stronger security, improved compliance with data regulations, and greater flexibility for cloud-based applications and remote users. Implementing ZTA within our MSaaS required an Identity and Access Management (IAM) system compatible with diverse infrastructures.

The Prototype leverages Keycloak, a mechanism to enable ZTA. Keycloak is an open-source software that acts like a security gatekeeper for applications. It streamlines user access by offering two key features: Single Sign-On (SSO) and IAM. With SSO, users only need to log in once to access multiple applications that trust Keycloak. IAM enables administrators to define user roles and permissions within each application. This ensures only authorized users can access specific features or data, keeping your applications secure. To enable true ZTA, an API Gateway was employed to manage access to all resources within the MSaaS implementation.

The API Gateway sits between the user and all user-facing resources. This service is intended to reduce the attack surface of the MSaaS application, by creating a common interface for user generated MSaaS requests. On each request, the user's authentication token signature is validated, additional security checks are also employed to check for revoked tokens on every level of data access. This gateway is intended to provide a first layer of security to all plugins and services integrated by customers or 3rd parties.

The Modeling and Simulation Applications serve as the front-end, or Presentation Layer, within MSaaS. To align with the principles of MOSA, the Presentation Layer must be fully abstracted from the underlying simulations. We employed a two-pronged visualization strategy utilizing both thin and thick clients. A thin client is a lightweight software application that depends primarily on a remote server for processing tasks. In contrast, a thick client, is a software application that performs most of its processing on the user's device, rather than relying on a remote server. This dual approach necessitates the use of common APIs to ensure interoperability between the different client types and supports two types of plugins: presentation plugins for the front-end, and server-side plugins for the back end.

For the thin client approach, Angular micro frontends are utilized as an architectural pattern. The micro frontend approach enables building large web application by splitting functionality into smaller, independent, and deployable units (i.e., plugins). For visualization within the thin client, both Cesium.js and OpenLayers are utilized. Cesium.js is an open-source JavaScript library for creating 3D globes and maps directly in the web browser leveraging WebGL. OpenLayers is an open-source JavaScript library to create interactive maps on web page. The combination of Cesium.js and OpenLayers gives a unique set of benefits: 1) it forces API consumption of server functionality in two unique visualization frameworks, 2) provides users with a 2D and 3D native environment based on their availability of WebGL and preferences and 3) provides the presentation layer a way to scale performance with available bandwidth without losing functionality. For the thick client approach, SitaWare, a 3rd party capability, and two CAE products, Ridge and Prodigy are utilized. SitaWare specializes in C2, Ridge provides an immersive VR experience, and Prodigy is an Unreal based (game engine) Image Generator (IG).

Apache Kafka is the middleware layer to enable communication between ABBs and Services. Kafka is an open-source event streaming platform that excels at handling real-time data. Kafka enables applications to publish data to the stream and others to subscribe and receive it as it flows. Importantly, Kafka also stores this data stream reliably, ensuring its persistence. This real-time data movement empowers applications to analyze and react to information in high-performance data pipelines, streaming analytics, and applications that require critical, up-to-the-moment information. As an example of Kafka performance, LinkedIn publishes the following metrics to its usage of its Kafka ecosystem. "Kafka ecosystem at LinkedIn is sent over 800 billion messages per day which amounts to over 175 terabytes of data. Over 650 terabytes of messages are then consumed daily, which is why the ability of Kafka to handle multiple producers and multiple consumers for each topic is important. At the busiest times of day, we are receiving over 13 million messages per second, or 2.75 gigabytes of data per second. To handle all these messages, LinkedIn runs over 1100 Kafka brokers organized into more than 60 clusters." (LinkedIn, 2024).

As a comparison, Traditional simulation systems operate at execution time frames ranging from 15 to 120 hertz. Using LinkedIn's Kafka performance metric, we can infer Kafka's suitability for handling simulation data. Assuming a simulation system executing at 60 hertz, Kafka's capacity suggests it can manage over 200,000 messages, and over 400 megabytes of data per simulation update over 1,100 Kafka brokers employed by LinkedIn. When normalized to a single broker, this translates to Kafka handling approximately 200 messages and 40 kilobytes per simulation update. Kafka offers significant advantages over traditional interoperability methods like DIS and HLA. Kafka's distributed architecture can handle a higher volume of messages and data, providing superior scalability. It ensures low-latency message delivery, critical for real-time applications, and includes built-in data persistence for reliable storage and fault tolerance. Because Kafka does not require a framework like HLA to define the data, it allows for easy integration with the wider variety data sources that will be needed when simulation contemporary operational environments. This makes Kafka ideal for modern simulation environments that demand robust data handling and seamless interoperability. In contrast, extant interoperability standards such as DIS and HLA, lack Kafka's scalability, flexibility, and data persistence capabilities.

RESTful (Representational State Transfer) endpoints, protocol buffers, and standards-based payloads conform to the MSaaS RA within the Kafka implementation. Simulation Services with similar architectures and interfaces are grouped into enclaves where member applications can communicate with common interfaces (i.e., DIS, HLA) via a traditional LVC Gateway to Kafka. For our implementation, we implementation a common data exchange model and simulation environment agreement ensure that applications in different enclaves can communicate across the cloud.

Kafka is utilized to implement the Mediation Service. Kafka Connect continuously pulls data from various sources (e.g., LVC Gateways, DIS, HLA, etc.) and pushes it into Kafka Topics (categories). Kafka Streams, a stream processing library, enables the implementation of custom connectors that process data as it flows through Kafka Topics to translate the data to the common data exchange (or mark the data as invalid). Fundamentally Kafka Streams perform transformations like filtering, aggregation, or data cleansing on the fly. Consumers with the "The Prototype" access Kafka Connect to access transformed data in its desired format. In essence, Kafka implements a many-to-one and one-to-many design pattern based on common data exchange standard.

Command and Control Systems to Simulation Systems Interoperation (C2SIM) is leveraged for "The Prototype" for common data exchange. C2SIM is a standard for expressing and exchanging Command and Control (C2) information among C2 systems, simulation systems, and Robotic and Autonomous Systems (RAS) in a coalition context. C2SIM enables a single message format for message logging, information visualization, data processing, data querying, knowledge reasoning, and other purposes across the whole coalition of systems. C2SIM covers the initialization, tasking, and reporting of forces. Initialization contains all information necessary for creating and describing forces, situation (weather, etc.), and control measures across interoperating C2 and simulation systems. Tasking and Reporting covers all information necessary to create tasks, provide situational reports, and manage forces between interoperating C2 system, simulation systems, and RAS. C2SIM deals with the exchanges of information among the types of systems listed above, to enable their collective initialization and operation as a coalition system of systems. (SISO-STD-019-2020).

In the implementation of C2SIM as a payload, we converted the C2SIM ontology described in SISO-STD-020-2020 into an equivalent set of schemas using Protocol Buffers. The translated protobuf schemas reference the combined C2SIM ontology, consisting of version 1.0.0 of the C2SIM Core ontology, with the C2SIM Land Operations Extension and C2SIM Standard Military Extension ontologies. The highest level of the ontology is the C2SIM Message, which is the form our protobuf payload takes. A Message has two main components: a header that gives general information about the Message, such as the communicating systems, the classification of the information contained in the Message, and the purpose of the Message; and a message body, which varies in structure depending on the purpose of the Message.

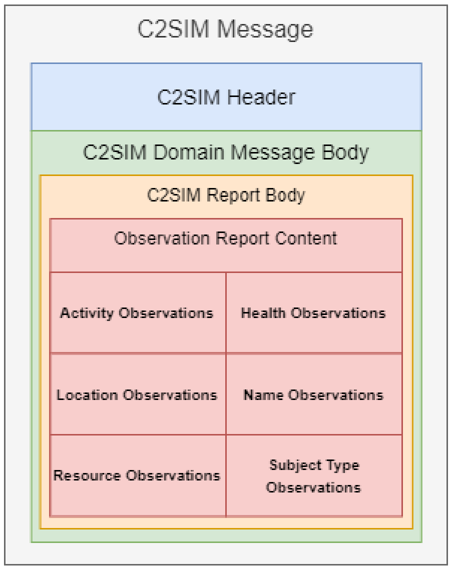


Figure 2: C2SIM Representation of State

An important note about C2SIM Observations is that, because C2SIM is meant for interoperation with C2 systems and thus for simulating real-world information passing between entities, all observations can fall within some margin of error and have a confidence level, To reconcile this with passing simulation information that is known to be perfectly accurate ("ground truth") between simulation systems and M&S, such as entity states coming from Computer Generated Forces sims, extensions have been proposed to differentiate Observations that are meant to reflect entity state coming from a source of absolute truth, and the canonical C2SIM Observations that simulate potentially inaccurate entity reports. An example of the hierarchy of the C2SIM Message payload for communicating entity state can be seen in the C2SIM Figure.

Other APs pertaining to M&S enabling services include the C2 simulation and distributed state data patterns in the common data exchange standard (i.e. C2SIM). The C2 simulation AP translates an event within the Modeling and Simulation Applications to an event understood by the simulations. C2SIM is the mechanism to transmit the order to perform Mediation Services between simulations. One of the experiments integrates intelligent agents into simulations and take natural language commands to drive the simulations through their APIs, thus partially replacing the need for traditional scenarios. This is discussed in the Scenario Service section.

Deploying a MSaaS RA in a scalable manager requires a mechanism to compose and scale simulations. Two methods are employed in "The Prototye": Kubernetes and vSphere. Kubernetes, often called K8S, is an open-source platform designed to automate the deployment, scaling, and management of containerized applications. Kubernetes manages where individual software components (containers) reside on servers, ensures they communicate effectively, and automatically scales resources up or down as needed. This orchestra tion enables efficient use of computing power, streamlined deployments, and continuous application health monitoring. Kubernetes into limited to a single environment, providing deployment flexibility for diverse needs. Cloud providers like Google, Amazon, and Azure offer managed Kubernetes services (GKE, EKS, AKS) handle the infrastructure and cluster management, letting users focus on applications. Alternatively, organizations can deploy K8S on their own hardware for maximum control, although this demands expertise in managing the underlying infrastructure. vSphere is a more traditional virtual machine approach enabling the encapsulation of monolithic applications. It manages the deployment and operation of virtual machines (VMs) on servers, enabling efficient resource utilization and simplified management of monolithic applications. It provides robust tools to monitor and maintain VMs, ensuring applications run smoothly and reliably. vSphere's architecture supports seamless scaling and resource allocation, adapting to changing workloads without manual intervention. With the ability to integrate with various storage and network solutions, vSphere offers flexibility in deployment environments, whether on-premises or in the cloud.

IMPLEMENTATION CHALLENGES & BENEFITS

Common Data

Common data in the M&S industry is an age-old problem. While standards exist for nearly every type of data to be exchanged, the implementation of standards in Simulation Services is a challenge. For various reasons beyond the scope of this paper, standards are good ideas, but good ideas do not always present valid business and technology cases for vendors and Governments to implement standards. The cornerstones of interoperability in LVC are shared order of battle, scenarios, terrain, event exchange, and entity control. While all the Simulation Services selected support DIS and/or HLA, these services do not share a Federation Object Model (FOM), or enumeration set. This challenge still exists within this MSaaS implementation.

MSaaS as a concept offers an abstract approach to achieve interoperability between Simulation Services. A concrete implementation is required for interoperability between Simulation Services. Therefore, a concrete implementation was established by defining the objects, attributes, and interactions as a concrete C2SIM definition. This required culling the complex ontology of C2SIM in a manageable and simple common functionality between the Simulation Services.

This requires the creation of custom plugins for MACE and VR-Forces to get the desired interoperability. These plugins served two functions: 1) they enabled raw access to raw entity data enabling a deeper integration to Kafka for publishing and subscribing to events between the various Simulation Service and 2) they enable dynamic injection of events – such as an order – for the Modeling and Simulation Applications (i.e. Presentation Layer). This provides tremendous optimization and efficiency for the user by granting the ability to inject natural language messages through the Presentation Layer that is automatically interpreted by the underlying Simulation Services.

For common terrain, OGC CDB is the foundational dataset format. OGC CDB enables global storage of terrain simulation data in non-mesh form. GeoServer dynamically streams OGC CDB to the Presentation Layer via OGC Web Service Standards. A Web Mapping Service (WMS) streams raster data, a Web Feature Service (WFS) streams vector data, and a Web Coverage Service (WCS) streams native content (i.e., GIS file formats) on demand to consumers. In the case of consumer requiring a prebuilt mesh (i.e., Cesium.js), a custom GeoServer extension produces 3D Tiles on demand for the Cesium.js customer. MACE and VR-Forces consume derivative forms of OGC CDB as stock GeoTIFF, Shapefiles, and DTED files. To achieve maximum automation, custom scripts automate the import of data from CDB into the 3rd party CGFs based on the user defined scenario playbox.

MOSA and Performance

Performance is the make-or-break concept of any simulation solution. As described earlier, the NDS outlines the need for large scale simulation solutions that operate at the campaign-level. It is paramount that future concepts of simulation effectively use cloud concepts to achieve scale. Historically, simulation solutions typically scale by clustering like CGFs into logically partitioned concepts like geography or domain on bare metal solutions. Performance in simulation is typically defined as objects per frame or update. In recent years the technology trend is leveraging 3rd party technologies that promise scale within their simulation platform.

With the advent of the "Platform Economy" success of companies like Salesforce, Amazon, Facebook, and many other Silicon Valley companies, the M&S industry is seeing a trend to follow suit and build proprietary platforms that enable vendors to vertically integrate within the M&S industry. To understand the concept of the "M&S Platform" allure, one must consider the elements of a platform. The general definition of a platform varies; however, the core elements of a platform include a common User Interface (UI) and User Experience (UX), Scalability and Performance, APIs, and specific value proposition.

The MSaaS RA describes the "platform" with respect to the SOA Services. MSaaS defines these SOA Services of a platform as security services, platform Service Management and Control (SMC) services (such as metering, monitoring, logging, and services discovery), message-oriented middleware services, composition services, mediation services, and information platform services (such as information discovery services and metadata repository services). Historically, it extremely difficult to treat the SOA Services as independent concepts from the MSaaS RA. In reality, SOA Services require a degree of tight coupling that are the antithesis of MOSA concepts.

The degree of coupling of the SOA Services and Simulation Services define performance. Kafka, vSphere, and K8 attempt the loosest possible coupling to the underlying SOA Services, to enable portability to other SOA Services. The MSaaS implementation was tested and deployed on the USAF SCARS On-Premise Equipment (OPE); however, it is equally important this solution work on commercial, public, or government certified clouds and on-premises equipment. The objective is to maximize adherence to MOSA concepts.

The M&S backend infrastructure connects the simulation service with Kafka through a gateway that translates the outgoing protocol packet into our canonical C2SIM representation. Each C2SIM Message has an average payload size upon conversion of around 240 bytes. In the case of MACE, for example, it translates DIS to C2SIM. MACE is a tick-based simulation service, and when operating at maximum simulation speed (x8), can send out around 450 states per second per entity. Load testing on our backend infrastructure was done on a system with a Core i7-10850H processor, and 64 GB of RAM. We found that our single-node Kafka messaging framework can support 750,000 messages per second on a single-partition, single-consumer instance. Our entity processing framework for ingesting incoming states was found to support around 15,000 C2SIM messages per second per entity, with a static memory footprint per entity of around 20 MB. As the above simulation services output an order of magnitude *less* states per entity, even at maximum supported simulation speeds, this means that our entity processing framework can support tick-based simulation services with much higher than real-time simulation speed and sets the groundwork for event-driven simulation support as well.

User Interface Performance

The modular thin-client web interface is intended to support visualization of data intensive LVC exercises. 2D geospatial visualization is built on OpenLayers an open-source mapping framework. 3D visualization uses Cesium.js. In the 2D view, over 10,000 entities can be visualized at once, and in 3D view, over 100,000 entities can appear on-screen. However, in most use-cases, this entity density provides too much data to draw insights from. To solve this challenge an intelligent clustering service is used to group entities based on order of battle (ORBAT), location, and other factors. We did not collect metrics for the thick clients.

Rapid Integration & Scaling

A clear benefit implementing MSaaS is rapid integration. The OPE from SCARS contains a SOA Service concept. This SOA Service concept made the implementation of M&S unique functionality highly flexibility. We used three different mechanisms to integrate Simulation Services: 1) creating VMs for CGFs, 2) creating containers for microservices, and 3) using commodity technologies like Kafka, vSphere, and K8 to scale and integrate Simulation Services. In retrospect, the consensus was anecdotal agreement processing scale was achieved in a simple manner using these approaches as compared to traditional methods. Furthermore, the on boarding time for services was in the scale of a few hours versus weeks of integration we traditionally encounter.

A unique element of the work was employing a global engineering organization on “The Prototype”. Developers (20+) were globally distributed (e.g., Texas, USA; Florida, USA; Montreal, Canada; Stolberg, Germany; Budapest, Hungary) contributing technologies to our MSaaS implementation. Using MOSA practices supported by C2SIM, WMS, WFS, WCS, and 3D Tiles as the common data exchange standards enables seamless collaboration between teams not co-located. Beyond the MOSA elements, the USAF SCARS OPE contains DevSecOps which significantly reduced integration and configuration management tasks.

Native Zero Trust Architecture (ZTA)

While the MSaaS RA does specify abstract concepts for security, building the M&S Composition Services with API Gateways in combination with Keycloak proved to be a powerful approach. Given a globally distributed development team, additional security mechanisms were required to ensure any export-controlled content was run in a controlled environment where developers from other sites had no access to it or its data. M&S Composition

Services provided this mechanism to enforce access control to functionality. This approach effectively leveraged Kafka topics to seamlessly segregate different data feeds based on authentication and authorization, ensuring that only authorized users could access specific data streams. Achieving this level of access control and data segregation in a traditional DIS or HLA environment would be significantly more challenging.

Extensibility

As a part of the environment, the creation of developer tooling enables the extension of the ecosystem. A CGF integration kit provides a foundation for enabling control of synthetic entities from inside the environment, additionally providing adapters for pushing data into Kafka over C2SIM. The user interface plugin template provides a starting point for creating new UI tooling, and visualization capabilities. Customer and 3rd party services can run within the M&S K8S cluster, and leverage the same native ZTA services, including SSO, gateway authentication, tokens signature verification. This plugin driven development approach utilizes the same resources as native functionality, ensuring maximum potential for future growth.

Discrete Simulation Functionality as Microservices

Individual microservices to perform discrete simulation functionality are required. Line of sight and movement services enable native scaling with K8S. Some algorithms such as line of sight might seem simple to move to a standalone microservice; however, the depth of dependencies and third-party frameworks produced significant complications. In some cases, it is easier to rewrite functionality with complex dependencies in lieu of encapsulating and severing dependencies. Scaling is another key lesson learned. Many traditional applications scale by multi-threading processes with shared memory. In a MOSA microservice environment, shared memory is not available. In essence, shared memory becomes a state of communication and messaging across the network or Kafka. Unintended second- and third-order effects of latency manifested where messaging bottlenecked processes. As a result, the ability to scale messaging frameworks horizontally is a paramount capability for MSaaS and MOSA simulation architectures.

The greatest benefit in transitioning functionality into a MOSA microservice environment is the speed to working software. A Bulkhead Pattern for architecture is an effective strategy to ensure the overall solution remains flexible and robust. Much like ships leverage bulkheads to create separate, watertight compartments to limit the impact of hull breach. the Bulkhead Pattern implies that each microservice is self-contained. This design ensures that a failure or performance degradation in an individual microservice does not compromise the entire solution. Achieving this requires deliberate and planned partitioning of functionality into logically hierarchical microservices.

CONCLUSION

An implementation of the MSaaS RA provides an opportunity to test for rapid application creation and scale for expansive LVC uses cases as outlined in the NDS. As is typical with a reference architecture, the MSaaS RA is an abstract architecture that requires concrete implementation. The concrete implementation defines key design decisions directly impacting the underlying tenants of MSaaS.

As a reference architecture, MSaaS defines a logical separation of services. Specifically, the reference architecture offers a clear distinction between SOA and M&S Services. While it is logical to define a clear separation between unique M&S and generic SOA capabilities, in practice the M&S Services are at risk of becoming tightly coupled to the selected SOA Services. The risk of MSaaS is that two different MSaaS implementations are incompatible because of potential differences of SOA Services. In implementation, the desire to be agnostic of SOA Services enable the portability of the solution to multiple SOA environments. The recommendation for future iterations of MSaaS consider provisions to ensure MSaaS implementations are agnostic with respect to SOA Services.

The hypothesis that MSaaS enables rapid applications and scale proved to be valid through a logical separation of services for applications creation – and more importantly integration. The approach of using Kafka as the “integration broker” proved to give significant flexibility and performance of integrating multiple Simulation Services together. Achieving scale with respect to processing is unfortunately highly dependent upon the SOA Service approach.

REFERENCES

Defense Standardization Program Office. (n.d.). Modular Open Systems Approach (MOSA). Defense Standardization Program. https://www.dsp.dla.mil/Programs/MOSA/

Department of Defense. (2019, January). Artificial intelligence and machine learning for rapid response [Technical Report AD1076559]. Defense Technical Information Center. https://apps.dtic.mil/sti/pdfs/AD1076559.pdf

Department of Defense. (2022, February 14). AI for wargaming application: A white paper [Technical Report AD1183539]. Defense Technical Information Center. https://apps.dtic.mil/sti/trecms/pdf/AD1183539.pdf

Ducharme, R., McAlexander, C., Mills, B., Freeman, J. (2023). Evaluation of open-source data for gray-zone operations decision systems. ResearchGate. https://www.researchgate.net/profile/Brian-Mills-11/publication/378288649_Evaluation_of_Open-Source_Data_for_Gray-zone_Operations_Decision-Systems/links/65d1191701325d46521179ab/Evaluation-of-Open-Source-Data-for-Gray-zone-Operations-Decision-Systems.pdf

Joint Chiefs of Staff. (2015, July 29). Dempsey: U.S. forces must adapt to deal with near-peer competitors. Joint Chiefs of Staff. https://www.jcs.mil/Media/News/News-Display/Article/613868/dempsey-us-forces-must-adapt-to-deal-with-near-peer-competitors/

LinkedIn Engineering. (n.d.). Running Kafka at scale. LinkedIn Engineering. https://engineering.linkedin.com/kafka/running-kafka-scale

NATO Science & Technology Organization. (2022). Adaptive autonomous systems: Current achievements and future trends [STO-MP-MSG-197-03]. NATO Science & Technology Organization. https://www.sto.nato.int/publications/STO%20Meeting%20Proceedings/STO-MP-MSG-197/MP-MSG-197-03.pdf