2021

Applying trade-space analysis to Modeling and Simulations as a Service (MSaaS)

A STUDY IN APPLYING ESTABLISHED SYSTEMS ENGINEERING METHODOLOGIES IN A NOVEL SETTING

NEIL KESTER

Neil Kester
13 May 2021
EN 645.802 SE Master's Thesis

Research Paper
Instructor: Dr. Strawser
Advisor: Dr. Montoya

Table of Contents

Table of Figures

Abstract

Techniques of trade-space analysis have been successfully employed for decades to support acquisition decisions. Likewise, large scale discrete event combat simulations have been successfully employed for a similar period to explore performance and effectiveness of systems and concepts considered in those trade-space analyses. Recently, great progress has been made breaking the paradigm of executing large and complex combat simulations as monolithic software systems run on local machines and primarily in serial. Teams like the North Atlantic Treaty Organization Science and Technology Office's (NATO STO) Modeling and Simulation Group have developed a novel way of serving these simulations as decoupled microservices in the cloud. They refer to this as Modeling and Simulation as a Service (MSaaS). The purpose of this paper is to demonstrate the suitability of integrating cloud based analytic resources into the immerging MSaaS field. It describes the composition of these cloud based analytic resources, extends NATO's existing MSaaS Reference Architecture to include these resources, and executes a simple trade-space design of experiments to demonstrate its utility.

Background

Trade-space analysis has long been a critical component of the Systems Engineering Process as it supports requirements analysis and architecture design processes. Conclusions from trade-space analyses inform decision analysis, technical planning, and technical assessment processes that follow. Often, computer simulations and mathematical models support trade-space analyses and allow systems engineers and design teams to identify the best systems, sub-systems, and architectures to meet specified user requirements. The process even informs the feasibility of functional requirements given constraining non-functional requirements such as cost, schedule, weight, power, etc. Unfortunately, these computer simulations and mathematical models often exist as monolithic systems that are hard to change or update, are limited by the computational resources of their static host system and are only accessible to those within the organizations that own or run them.

In contrast, the concept of Modeling and Simulation as a Service (MSaaS) leverages cloud resources and distributed architectures to make computer simulations and mathematical models accessible to many users. These models and simulations run atop scalable computational resources and can easily change to fit specific analytic needs. The adoption of Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) solutions have empowered individuals and organizations to manage only the level of complexity they require and scale resources to meet their demand. In the same way, MSaaS allows users to select various models and simulations, compose them into larger simulations, and execute them to accomplish their objectives.

More completely, the North Atlantic Treaty Organization (NATO) Science and Technology Office (STO) Modeling and Simulation Group (MSG) 164 defines MSaaS in the following way:

"An enterprise-level architecture that promotes modularity, loose coupling, agility, and reusability of Modeling & Simulation resources for different suppliers by making them available on-demand to a large number of disparate users in order to reduce

the cost and time for implementing Modeling & Simulation capability to improve operational effectiveness." [1]

Of the NATO MSaaS user communities, those focused on training and operations currently dominate its development priorities. That focus is appropriate as it shows value to leaders in those nations and enjoys a larger user base. A byproduct of that focus on the training community is that much room exists for development and improvement when applying MSaaS to the analytic community.

As such, the identified problem this research intends to address is that NATO's MSaaS lacks an integrated analytic framework capable of supporting concept development and analytic support to procurement decisions. It should be said that these analytic capabilities could benefit the training and operations communities as well as the analytic community. [2]

Study Purpose

Given that context and the initially described gap, the purpose of this study is to determine if Modeling and Simulation as a Service (MSaaS), a novel approach to traditional Modeling and Simulation solutions, is suitable for conducting trade-space analyses. These trade-space analyses support systems engineering questions during the requirements analysis and architectural design technical processes of the US DoD Systems Engineering Process.

Study Hypothesis

The author's hypothesis is that a NATO MSaaS instance, integrated with an analytic environment, is suitable to support trade-space analysis by measuring the robustness of military systems.

Study Goals

The study has two goals that relate to that hypothesis:

1) To extend the existing NATO MSaaS framework with an analytic environment.
2) To execute a simple trade-space analysis using an existing NATO MSaaS scenario and the analytic environment previously mentioned to demonstrate the suitability of this approach to support this type of work.

Study Questions

To prove or disprove the hypothesis, this study will answer the following questions:

1) What are the challenges with integrating an MSaaS instance with an analytic environment?
2) What information model supports communications between the MSaaS models, simulations, data stores, and analytic tools to support trade-space analyses.
3) How could integrating MSaaS with an analytic environment support trade-space analyses?
4) What are the benefits and risks of this approach?
5) What are the limitations of this approach?

Materials & Methods

Methodology

To answer the study questions and achieve the study goals, work was divided into five methodological steps as shown in Figure 1: Methodology. Each step requires inputs and produces outputs. These outputs are either used by subsequent steps or are artifacts that serve as outputs from the study.

These methodological steps are mapped to the previously described study questions, ensuring the work furthers research required to prove or disprove the study's hypothesis. This methodology traceability matrix is show in Figure 2: Traceability Matrix.

Through the rest of this section, I will walk through each of the methodological steps, describe the work completed in each and how that work related to the study questions.

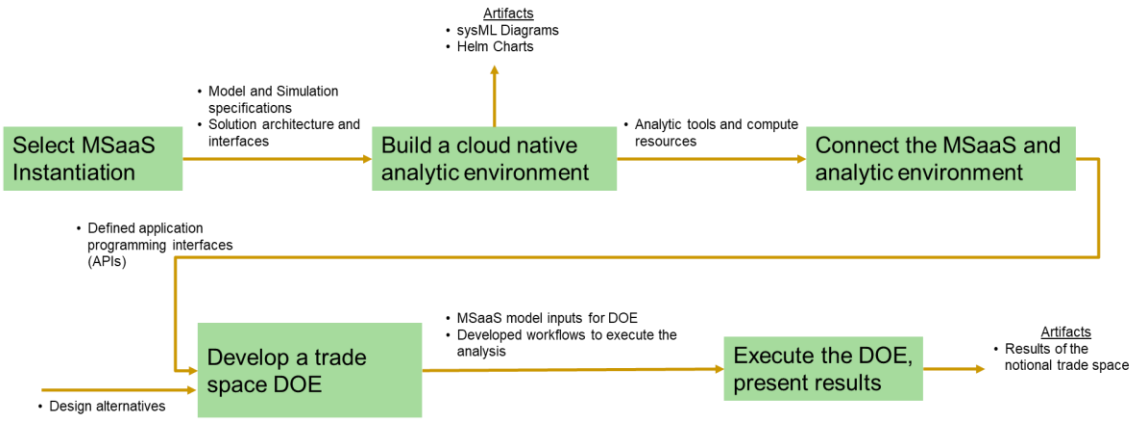


Figure 1: Methodology

	Select MSaaS Instance	Build a cloud native analytic environment	Connect the MSaaS and analytic environment	Develop a trade space DOE	Execute the DOE, present results
What are the challenges with integrating a MSaaS instance with an analytic environment?	X	X	X		
What information model supports communication between the MSaaS models, simulations, and data stores and analytic tools to support trade-space analyses	X	X	X		
How could integrating MSaaS with an analytic environment support trade-space analysis?			X	X	X
What are the benefits and risks of this?	X	X	X	X	X
What are the limitations of this?	X	X	X	X	X

Figure 2: Traceability Matrix

Step 1: Select an MSaaS Instantiation

The first step in the methodology is to select a Modeling and Simulation as a Service (MSaaS) instantiation. The North Atlantic Treaty Organization Science and Technology Office's (NATO STO) work on MSaaS has produced a reference architecture and foundational guidelines to establish and maintain an MSaaS instance. Each implementation of this reference architecture differs in where it is deployed (what cloud service provider), how it is secured, and how it is networked, among other implementation decisions. Due to familiarity, existing relationships, and mutually beneficial work, this study used an MSaaS instance implemented by the U.S. Army's Night Vision and Electronic Sensors Directorate (NVESD).

Leveraging NVESD's research MSaaS provided the added flexibility of hosting the analytic resources within the same cluster as the MSaaS. This greatly reduced the overhead of creating an independent Kubernetes environment. It also reduced potential configuration steps required to communicate between separate clusters. Finally, this instance has active developer support whose work aligned with this study's objectives. This symbiotic relationship aided in building additional capabilities within the existing MSaaS simulation composition to support this work.

To address study question #2, what information model supports integration with NATO's MSaaS work, I extended the Modeling and Simulation Group's (MSG) existing use case diagrams and sequence diagrams to include these resources. The products of that work are included in Appendix 1: Information Model. In short, it adds a new entity, the "Analyst" to the MSaaS high level Use Case Diagram. That "Analyst" entity is described by several subordinate Use Cases that include topics ranging from submitting new analytic resources for registration in the MSaaS instance to addressing the analytic objectives of a simulation event.

Step 2: Build a cloud native analytic environment

Goal #1 of this research is to extend the existing NATO MSaaS Reference Architecture to include analytic resources. To make the products from this research as interoperable as possible, I adopted NATO's

concepts of architectural building blocks (ABBs) and architectural patterns (APs), to describe potential analytic resources. The analytic environment can be described by three types of ABBs: integrated development environment (IDE), analytic engine, and database management system (DBMS). These three types of ABBs can be combined in any number of ways into architectural patterns (APs). The three ABBs and an example of potential APs are shown in Figure 3: Analytic Environment as a Service (AEaaS) Architecture Ontology.

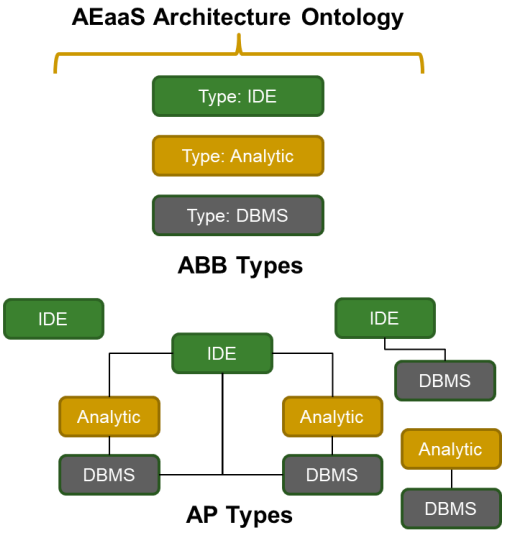


Figure 3: Analytic Environment as a Service (AEaaS) Architecture Ontology

From that reference architecture, I selected an architecture that included one IDE: RStudio, two analytic engines: R and Python, and two database management systems: MongoDB and PostgreSQL. To implement that AP, I customized existing software containers from projects rocker[1] and bitnami[2] and integrated their Kubernetes deployment specifications in Helm charts. This approach allowed me to leverage existing work, customize it to my requirements, and dynamically manage the version and deployment of each resource. Artifacts from that process were stored in dockerHub and GitHub and were deployed into the MSaaS environment described in Step 1. [3] Figure 4: Analytic Resource Workflow graphically depicts the previous text description.

[1] Provided by the rocker project at https://www.rocker-project.org/
[2] Provided by bitnami at https://bitnami.com/
[3] These resources can be found at the following projects:
https://hub.docker.com/repository/docker/neilkester/thesis-rstudio https://github.com/nkester/Thesis-Charts

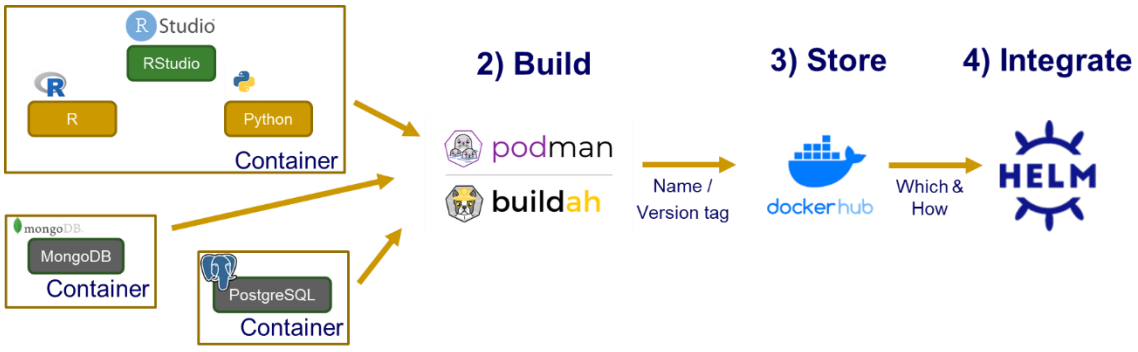


Figure 4: Analytic Resource Workflow

Once those ABBs (containers) and AP (helm chart) were built, tested, and stored in dockerHub and GitHub, I deployed them into the MSaaS Kubernetes environment. That deployment action concluded Step 2 of this study's methodology and allowed me to work on integrating the two systems in Step 3.

Step 3: Connect the MSaaS and analytic environment

Due to the decision in Step 1 to deploy the analytic environment in the same cloud instance as the MSaaS, connecting the MSaaS and analytic environments was fairly easy. The interface between the two was simply the previously described MongoDB where the simulation would save its outputs and log data. While other connections between the MSaaS and analytic environments are ideal, such as configuring and launching simulation runs, those connections are beyond the scope of this work. For this research, those configuration and initiation actions occurred through a web interface and sufficed the goal to demonstrate suitability.

Step 3 concluded when the MSaaS simulation demonstrate it could connect to the MongoDB and save simulation results. **This capability completed step 3 of the methodology and also achieved Goal 1 of the research, to extend the existing MSaaS framework with an analytic environment.** In achieving this goal, I, playing the role of the analyst, interacted regularly with Dr. Robert Kewley who served the role of the MSaaS integrator (see Appendix 1 for the MSaaS Information model and a deeper description of these use cases).

Step 4: Develop a trade space DoE

The selected MSaaS instance had an existing discrete even combat simulation composition that could be enhanced to support this work's analytic objectives. The composition included a terrain model, a service that returned line of sight results, and an implementation of the acquire sensor acquisition model among other services. The Acquire model determines if a sensor can detect another entity in the simulation given environmental conditions like light level, visibility, and thermal clutter. Those capabilities provide all of the major modeling components required to support a simple trade-space analysis and design of experiments.

The focus of this work was not to develop a complex or efficient design of experiments. Instead, it intends to show suitability. In that view, I developed a design of experiments that took the highest and lowest categories of light level and visibility parameters in the acquire methodology as the treatments. High (+) for light level represented mid-day light. Likewise, the low (-) level for light level represented a

moonless night. The high (+) level for visibility was "clear" and the low (-) level was "hazy." The experimental design was therefore all four combinations of those treatments.

Likewise, sensors come in many forms and capabilities but can generally be categorized by unaided human eyes, night vision, and thermal. To develop systems under test that should show significant performance difference, I choose System A to be unaided eyes during the day and night. I chose System B to be binoculars during the day and performant night vision goggles at night. Finally, I chose System C to be a helicopter's day sights during the day and a performant forward looking infra-red (FLIR) sensor at night.

Those experiments and systems combined into the following 12 design point run matrix shown below:

Environmental Conditions:

Experiment	Light		Visibility		Iterations	Parallel	Run Time (min)
1	+	Clear Daylight	+	Clear	3	3	8
2	-	Clear Moonless Night	-	Hazy	3	3	8
3	+	Clear Daylight	-	Hazy	3	3	8
4	-	Clear Moonless Night	+	Clear	3	3	8

Systems Under Test:

System	Entity	Sensor Package	Day Sight	Night Sight
A	WASP 1	"vehicle passenger"	"eyes"	"eyes"
B	WASP 1	"vehicle commander"	"binoculars"	"nvg"
C	WASP 1	"uav sights"	"heloDaySight"	"hunterFLIR"

Design Point Run Matrix:

Design Point	System	Experiment
A1	A	1
A2	A	2
A3	A	3
A4	A	4
B1	B	1
B2	B	2
B3	B	3
B4	B	4
C1	C	1
C2	C	2
C3	C	3
C4	C	4

The simulation provides as an output, the line-of-sight state for every entity-to-entity pair when that state changes. Likewise, it also provides the acquisition state (e.g. not detected, detected but not identified, etc.) at each state change event. The simple metric used then to determine how effective a sensor was at acquiring targets was the difference between the number of targets the entity has the ability to see (that was in its line of sight) and the number of targets the sensor actually acquired.

Of note: Any level of detection was counted as an acquisition for the purpose of this study.

Since robustness is a measure of the effectiveness of a system, even in adverse conditions, the most robust sensor is the one that minimized the total "missed acquisitions" across the experiments. [4]

Step 5: Execute the DOE and present results

The final step of this study's methodology is to execute the DOE and use the analytic environment to perform a trade-space analysis on the results. Each design point was configured through the MSaaS simulation's web interface which specified which scenario to run (which system was being evaluated)

[4] Cite WIPRO at https://www.wipro.com/blogs/girish-datar/robust-or-resilient/#:~:text=Resilience%20is%20the%20ability%20to,state%20of%20performance%20or%20better.

and what the environmental conditions were (light level and visibility). Additionally, the researcher specifies the number of iterations to perform and how many should be conducted in parallel.

Due to the research nature of the selected MSaaS instance and cognizance of the cost of the work, this research ran each design point for three iterations and conducted all of them in parallel for the record runs. In testing, however, design points were run for 100 iterations with 10 and even 15 in parallel. This demonstrates the scalability of this approach given large or important analyses that require demonstration of statistical significance of the results. The ability to scale these simulation execution horizontally like that demonstrates this approach's suitability to support trade-space analysis.

As the MSaaS simulation completed its design point iterations, it wrote results to the analytic environment's MongoDB. As MongoDB is a No SQL document store, to conduct analysis on the results, pertinent elements must first be retrieved, transformed, and related to each other. From there, the results could be analyzed, and results presented.

To deal with large volumes of structured and related data, I leveraged the PostgreSQL DBMS previously described to store and serve those data. This again demonstrates the suitability and flexibility of leveraging MSaaS and cloud resources for this type of work. Had the data or analytic questions been better suited if served by a graph database, that could easily have been deployed in place (or in addition to) the PostgreSQL database. Regardless, the ability to leverage dynamic, adhock DBMS solutions allows analysts to query and structure the raw simulation results once for each design point but query said data repeatedly and efficiently from PostgreSQL.

I developed a custom package in R, an open-sourced statistical computing language, to encapsulate each step of this process. The package, called "{modSim}", has many internal functions but exports only a handful for use by the analyst to simplify implementation. Encapsulating workflows in this way helps manage change and inherently makes the processes more transportable.

The functions available from "{modSim}" allow an analyst to follow the traditional "extract, transform, load" (ETL) workflow described above to query from MongoDB (extract), structure and clean (transform), and write to PostgreSQL (load) simulation results. In addition, it provides functions to query the PostgreSQL database, analyze the data, and visualize the results. An artifact like this package would be considered something that could be registered as an analytic service and be included in an MSaaS simulation composition. This would provide future analysts a way to extract and begin analysis of future implementations of this simulation composition. Instructions and a demonstration of how to use the "{modSim}" package is included in Appendix 2: Using `{modSim}`.

This methodological step achieved Goal 2 of this research.

Results

This section focuses on the results attained by completing the two specified research goals. This describes the results in more detail.

Goal 1: Extend MSaaS with an Analytic Environment.

In order to extend the existing MSaaS reference architecture with analytic tools and to support analytic workflows, I developed new architectural building blocks and architectural patterns. The following

sequence of diagrams shows the progression from NATO's MSaaS architecture framework to the specific analytic tool architecture I implemented. The results of this work will feed back into the MSaaS Modeling and Simulation Group's work and be added to their future reference architecture.

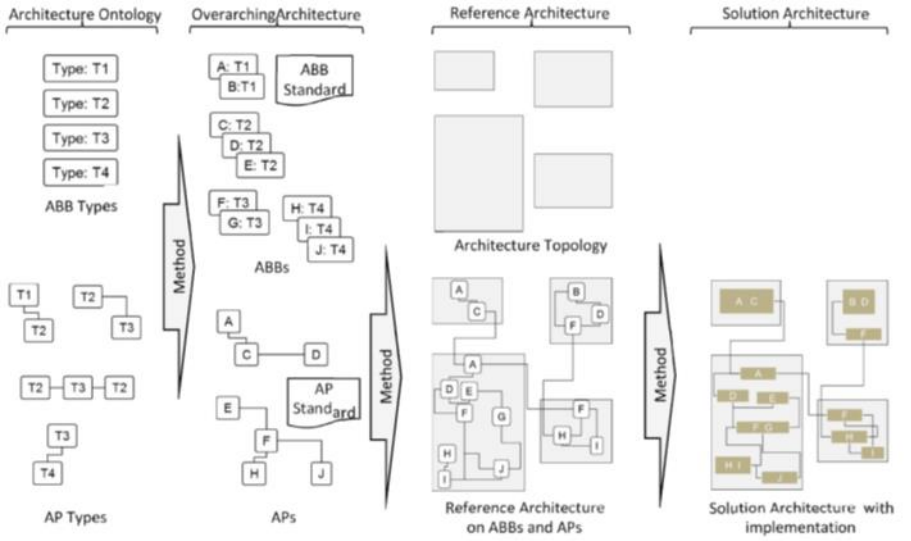


Figure 5: NATO MSaaS Architecture Framework

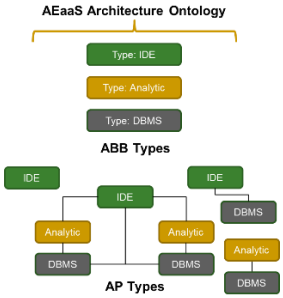


Figure 6: Analytic Architecture Ontology

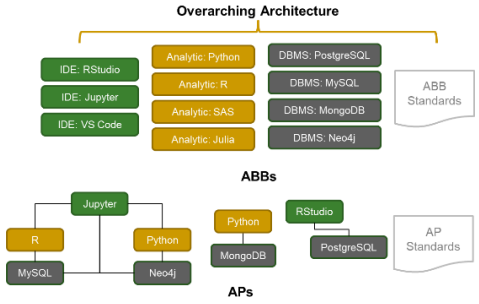


Figure 7: Overarching Architecture

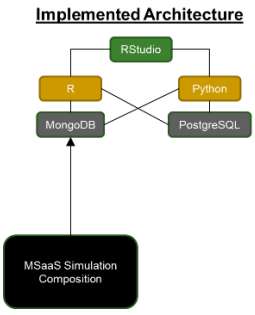


Figure 8: Implemented Architecture

The architecture alone or even building the components of the architecture, however, is not enough to extend MSaaS with an Analytic Environment. Those components must be physically deployed and able to connect to each other as well as the MSaaS they are intended to extend. Figure 9: The Final Product is a depiction of the architecture used to support this research. The "Multi Domain Operations Sim" box and the MongoDB from which it receives input data is an oversimplification of the MSaaS Simulation Composition used to support this work but is all that is required for this purpose. It connected to the elements of the analytic environment through a second MongoDB as shown in Figure 8: Implemented Architecture above.

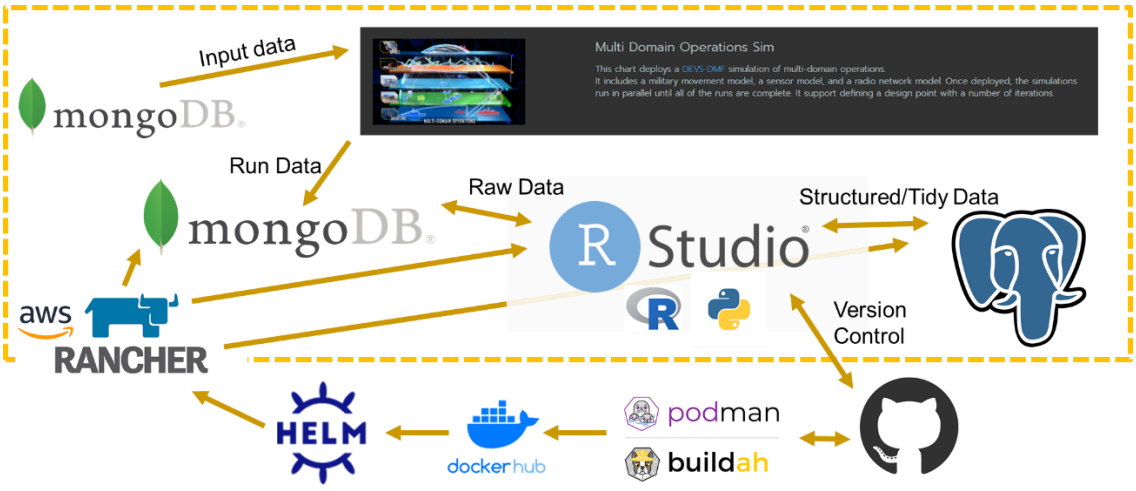


Figure 9: The Final Product

Goal 2: Execute a simple trade-space analysis.

This research never intended to develop a complex design of experiments or make a meaningful conclusion from a complex and in-depth trade-space analysis. Instead, it intended to demonstrate the suitability of using MSaaS, extended with an analytic environment, to conduct analytic workflows like conducting trade-space analysis.

Regardless, Table 1: Environmental Conditions, Table 2: Design Points, and Table 3: Systems Under Test constitute the simple design of experiments used in this research. The 12 design points resulting from this experimental design were run in the MSaaS Composition and produced results shown in Figure 10: Missed Acquisitions by Experiment and System.

Table 1: Environmental Conditions

Experiment	Light		Visibility		Iterations	Parallel	Run Time (min)
1	+	Clear Daylight	+	Clear	3	3	8
2	-	Clear Moonless Night	-	Hazy	3	3	8
3	+	Clear Daylight	-	Hazy	3	3	8
4	-	Clear Moonless Night	+	Clear	3	3	8

Table 2: Design Points

Design Point	System	Experiment
A1	A	1
A2	A	2
A3	A	3
A4	A	4
B1	B	1
B2	B	2
B3	B	3
B4	B	4
C1	C	1
C2	C	2
C3	C	3
C4	C	4

Table 3: Systems Under Test

System	Entity	Sensor Package	Day Sight	Night Sight
A	WASP 1	"vehicle passenger"	"eyes"	"eyes"
B	WASP 1	"vehicle commander"	"binoculars"	"nvg"
C	WASP 1	"uav sights"	"heloDaySight"	"hunterFLIR"

Figure 10: Missed Acquisitions by Experiment and System shows the sensor's missed acquisitions by time step across the four experiments and three sensor packages. This metric of missed acquisitions is simply the difference for each time step between the number of enemy targets the sensor had the ability to see (had line of sight with) and how many it did "see" (had some level of acquisition with).

This metric is useful because it allows the analyst to evaluate how robust a sensor's performance is in the face of adverse conditions, in this case varying light and visibility levels. With larger run sets (more iterations), the analyst could make statistical conclusions as to whether a sensor's performance during one experiment was different, with some level of statistical significance, from a baseline experiment. The analyst could also make the conclusion if one sensor performed better than another sensor during an experiment or even if one sensor was in fact more robust (better performance across all experiments) than another sensor in the analysis.

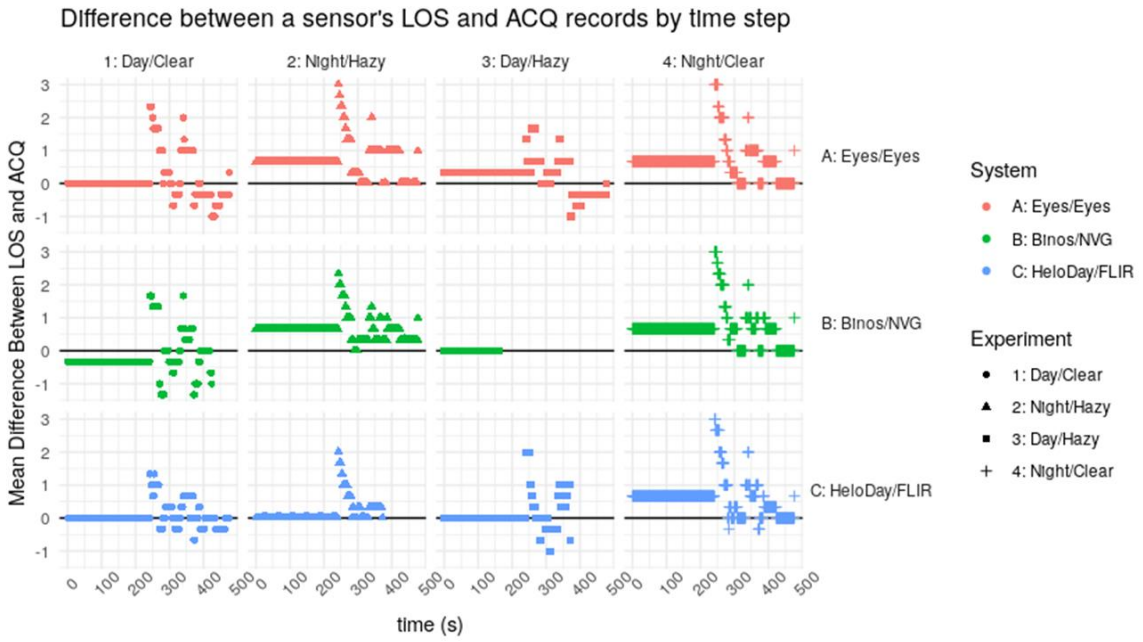


Figure 10: Missed Acquisitions by Experiment and System

Conclusion

Based on the results from Goals 1 and 2 described in the previous question, this research concludes that MSaaS, integrated with an analytic environment, is suitable to support trade-space analysis. While this research did not produce a metric or measure that proves this suitability, it demonstrated the utility of the approach. It is my hope that future research will continue this work. The next step is to compare the performance and suitability of conducting a full scale trade-space analysis on traditional, monolithic software simulations as compared to an environment similar to what was presented in this paper.

Acknowledgements

Acknowledgements here

Appendix 1: Information Model

NOTE: ADDITIONS IN *ITALICS* MADE BY NEIL KESTER EXTENDING THE INFORMATION MODEL FOR ANALYTIC USE CASES.

Use Case Actors

There are three main types of users of an MSaaS solution with different responsibilities. Each is described below.

Simulation Operator

The Simulation Operator is responsible for a correctly functioning simulation deployment in the MSaaS solution. This includes (but is not limited to) the verification that the various (simulation) services function properly, that is, with compliance to agreed service interfaces and contracts; the planning for and allocation of computing resources for a simulation deployment; the start and termination of a simulation deployment; the monitoring and control of a deployment, the configuration of simulation scenarios.

This Simulation Operator should be able to view or search available compositions, deployments, and scenarios that can be executed. The Simulation Operator should also be able to make decisions about how particular service will be deployed, e.g. on what computing resource and at which service port.

The Simulation Operator is knowledgeable in both the simulated environment as well as the reasons the simulation environment is used (analysis, experimentation, testing, training, etc.).

The Simulation Operator works with the Analyst when browsing compositions intended for analysis uses to ensure the composition appropriately addresses analytic objectives.

Integrator

The Integrator is familiar with the M&S service execution and configuration from a "black box" perspective (not necessarily having access to the internal aspects of the service). The Integrator understands how to deploy and execute the M&S service including how to manage its configuration and parameters. The Integrator also understands the capabilities of a collection of M&S services to manage compositions of multiple M&S services for the execution of scenarios. The Integrator builds compositions, to be used by the Simulation Operator along with defining all the configuration options the composition affords, to be used by the Simulation Operator. The choices ultimately made by the Simulation Operator will dictate the configuration of the M&S services as well as the deployment details (i.e. where to deploy the M&S services).

The Integrator works with the Analyst when building compositions with analytic services or analytic objectives. This coordination ensures the tools, methods, and approaches in the select analytic services integrate with selected M&S services and are appropriate for the analytic objectives.

Supplier

The Supplier is the source of individual M&S services. The Supplier is the technical expert on their respective services, including the underlying syntax (service interface and protocol details), semantics (how capabilities are represented within the service), and pragmatics (how the service can interact

within a composition of services). The Supplier needs to provide the service implementation, but also provides information about configuration options and mechanisms to use the service implementation in multiple ways including: scenarios, functional representation options, technical integration environments, and everything else required to allow the Integrator to use the Supplier's service in varying compositions. The Supplier also needs to provide license information, security information, and deployment restrictions.

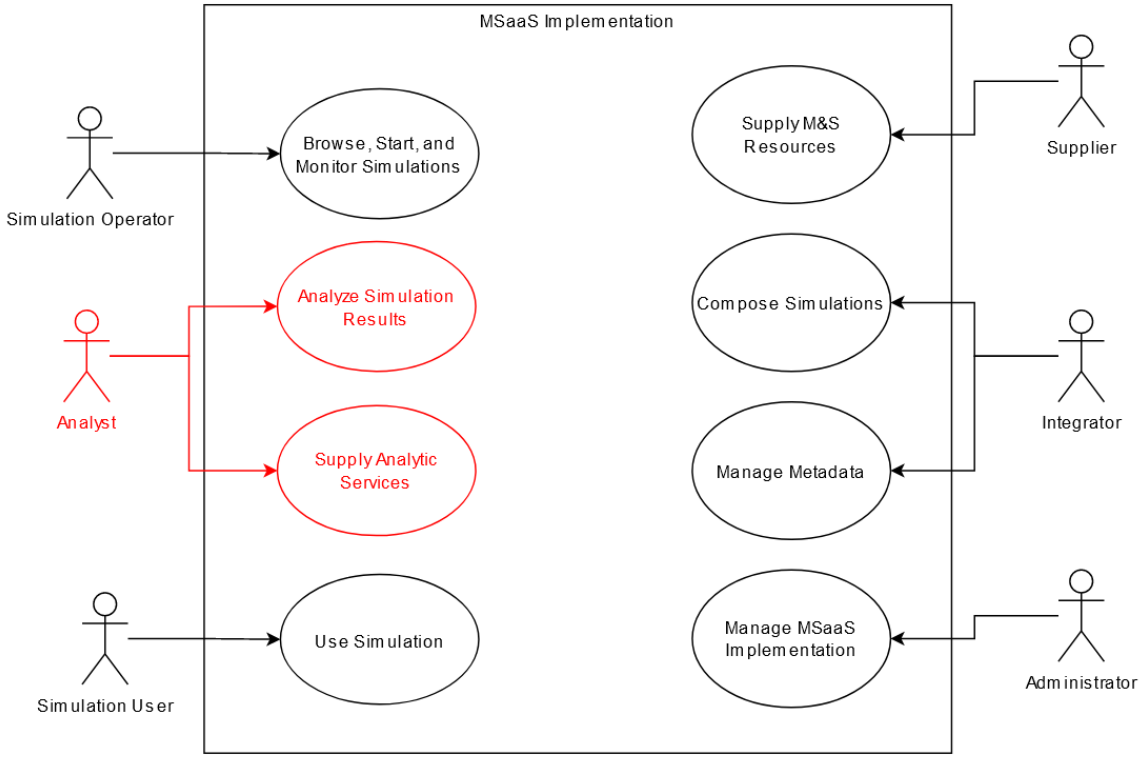
Analyst

The Analyst is the source of individual analytic services and deployed analytics. Analytic services and deployed analytics differ in the function they serve for the composition. In general, the analytic service supports the data exploration, analysis process, and experimentation required to answer a study question. Conversely, a deployed analytic is the result of that scientific process and provides the results of an analytic workflow to Simulation Operator(s) provided known data sources study questions. In other words, the analytic service allows an analyst to determine what chart to show, what data sources, colors, and labels to use while the deployed analytic shows that completed chart given the results of the composition's execution.

As the source of analytic services and deployed analytics, the Analyst works with the Integrator to supply complete configuration information and metadata for the service.

Likewise, the Analyst works with the integrator when browsing services to create new compositions. The Analyst is familiar with analytic services, the methods they employ, and the tools they use, and the assumptions and limitations described in their configuration information and metadata. This understanding allows the Analyst to assist the Integrator when building compositions with analytic objectives.

Finally, the Analyst works with the Simulations Operator when browsing existing compositions. The Analyst verifies the analytic services and/or deployed analytics function properly with the composed M&S service(s). Like the Simulation Operator, this include, but is not limited to, planning for and allocating simulation computing resources and the monitoring and control of deployed analytics and their results. In this role, the Analyst ensures the deployed analytic provides reliable and accurate results to the customer.



- Note: Red use cases indicate additions to NATO's MSaaS Information Model made by Neil Kester

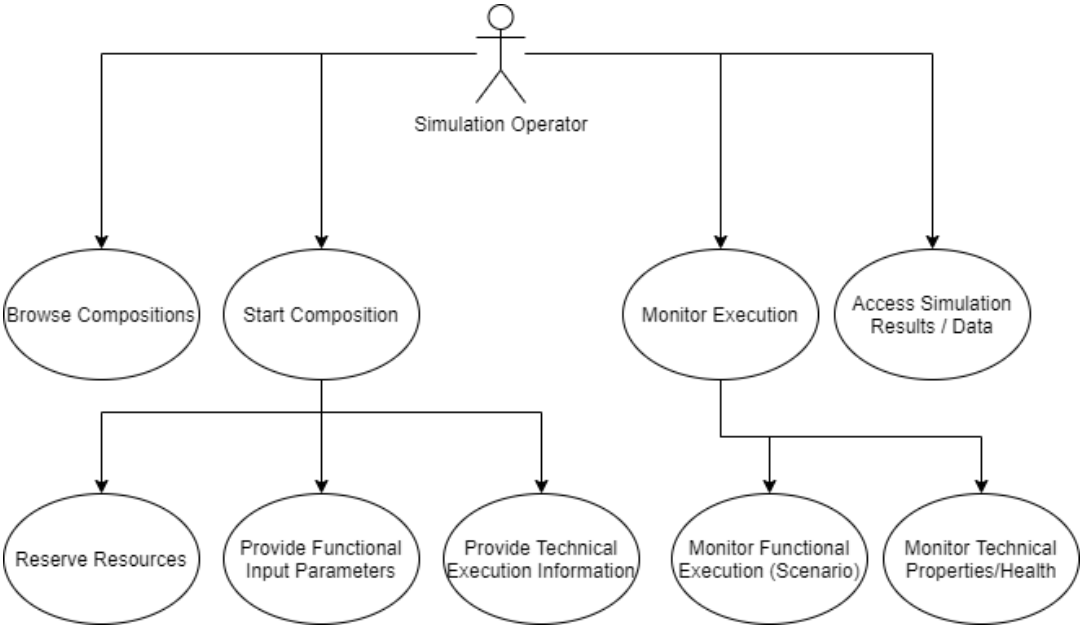
Use cases

Use Case Template

Use cases within this document abide by the structure below including the information within the table.

Use Case Element	Definition
Name	Name of the Use Case
Actor	The role that performs the defined behavior
Triggers	Events that cause the use case to be initiated
Pre-Conditions	Conditions that must be met before this use can start
Post-Condition	What the system should have completed by the end of the steps

Simulation Operator use cases



UC-1: Use Simulation

Objective: to execute a simulation, which includes browsing existing composition and deployment descriptions; starting, monitoring and terminating a simulation execution; accessing data from past simulation executions; and working with Suppliers and Integrators to create, configure, or modify composition and deployment descriptions. This use case incorporates the other use cases within this section.

UC Actor: Simulation Operator

UC Triggers:

- Operator: A simulation event exists and requires a composition of M&S Services to be executed at the event.

Pre-Conditions:

- M&S Resources – including composition and deployment descriptions - are provided by the M&S Repository Services
- M&S Resource Metadata is provided by the M&S Registry Services
- The MSaaS Portal Applications provide access to the M&S Resource Metadata

Post-conditions:

- Composition and deployment descriptions created for the simulation event
- Specific simulation event-related configurations of deployment descriptions created
- Simulation executed, and data is available during and after the execution has completed

UC-1-1: Browse Compositions

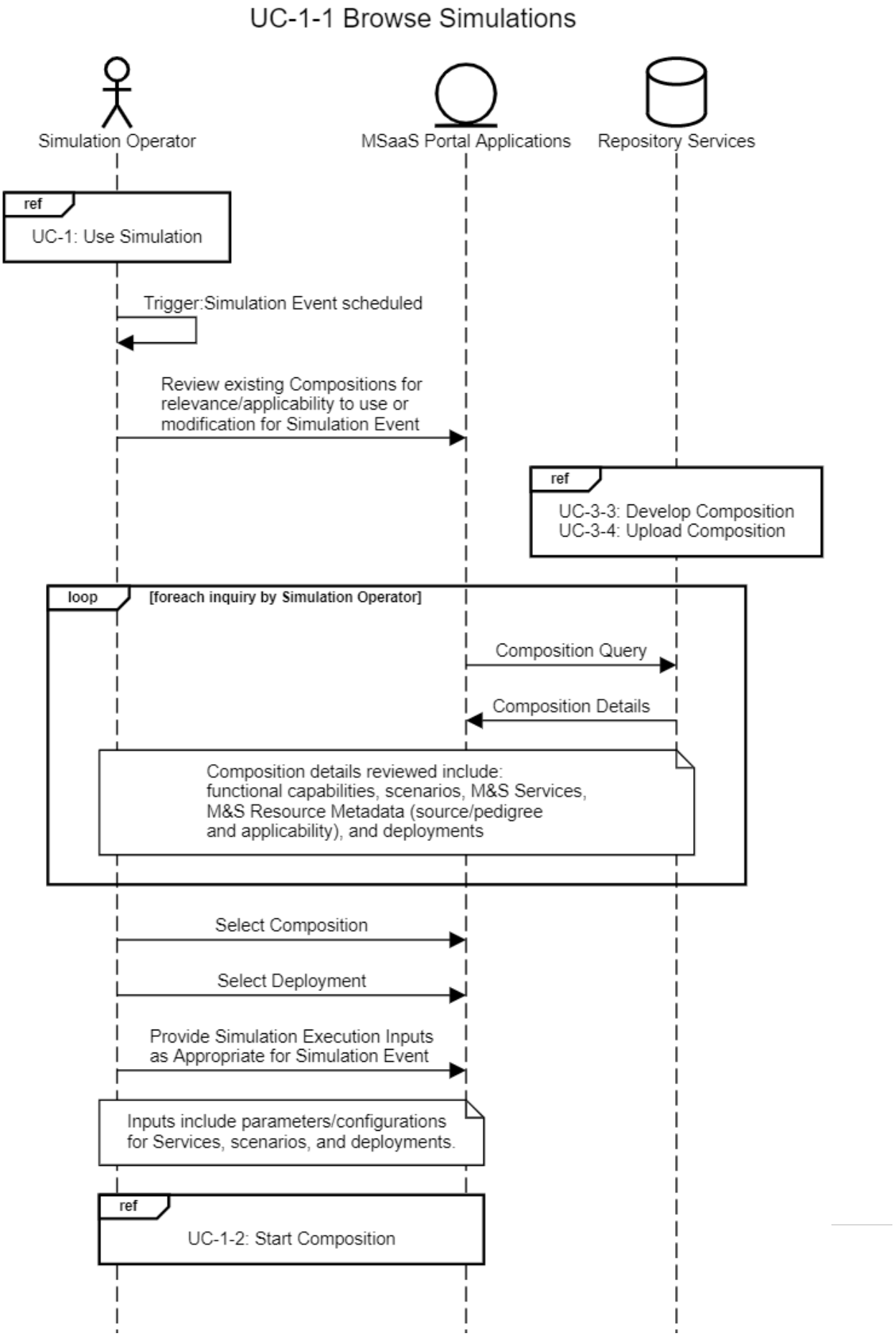
The Simulation Operator will require a user interface to explore and/or search for composition and deployment descriptions to execute for the simulation event.

Assumption: It is likely that no existing composition or deployment description will exactly match what the Simulation Operator needs for the simulation event. It is more likely that the Simulation Operator will need to work closely with the Integrator and possibly Suppliers to create custom configurations of deployment descriptions, scenario descriptions (e.g. Military Scenario Definition Language [MSDL] file), and possibly new functionality for the simulation event. The M&S Implementation may facilitate this workflow, but this information model assumes that the conversations between the Simulation Operator and the Integrators/Suppliers occur offline and just the results of their work would then be captured within the MSaaS implementation.

When attempting to find a composition or deployment, and execute a simulation, the Simulation Operator will be viewing information about compositions/deployments and their linked M&S Resources (including M&S Services, Scenario descriptions, and associated metadata). Each composition could execute many different scenarios as long as the configuration of the deployment is either not tied to the scenario or is dynamic in nature so that the scenario can be injected. The Simulation Operator will need to see the following details about compositions (to be included in the composition description):

- Functional capabilities of the composition describing what is represented in the simulation, and to which level of detail:
    - These capabilities could be simply keywords (e.g. medical supplies) or they could be more detailed and based on a central (domain) taxonomy (or an ontology) that enforces common vocabulary and denotes relationships among functions.
    - Capabilities could include:
        - a link to specific term/meaning within the functionality taxonomy/ontology
        - pedigree (source and approval) of the models
        - fidelity – accuracy (to be determined how to represent this in a standard way)
        - resolution – detail (to be determined how to represent this in a standard way)
        - Additional metadata including model limitations, assumptions
    - Information provided by the Supplier
- Related simulation scenario(s):
    - Location – place in the world that the scenario is occurring and/or if the location is fictional, then providing some geo-typical information such as a desert, forest, etc.
    - Entity information
        - Force laydown / initial positions
        - Force types (e.g. future force: a proposed future force structure)
        - Planned movements / actions
        - Entity types and count

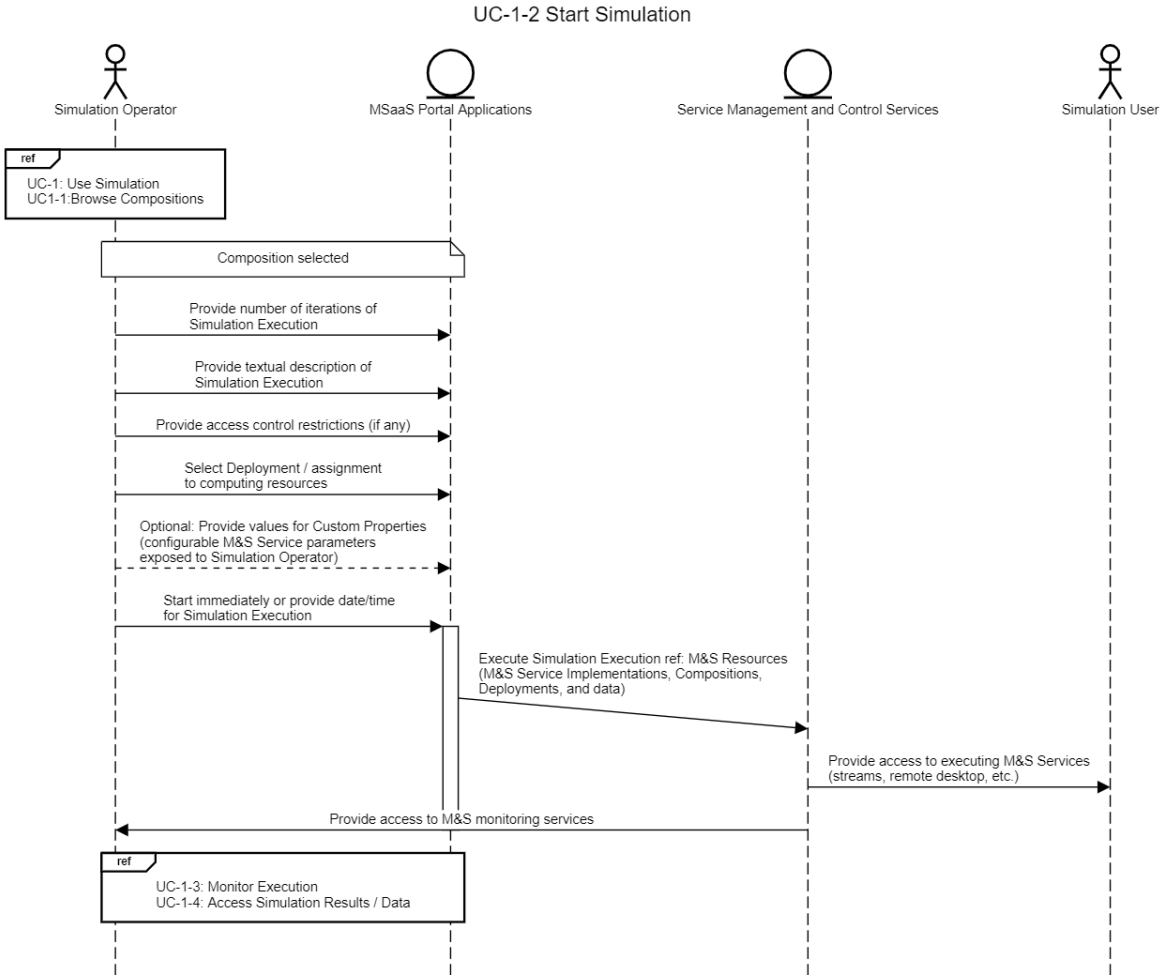
        - Unit aggregation when possible. For example, saying that a Brigade is at a location rather than showing where every single entity is would be easier to view unless the user wants to zoom in with that level of detail
    - Information provided by scenario developers and assigned by the Integrator
- Dependencies on external systems and services:
    - Although it would be ideal if the user was only concerned about functionality, this can be important when the user is required to use a specific (external) system for their simulation event. For example, C2 systems.
    - Information provided by the Integrator
- Data
    - Description of the data used by the M&S Services to execute the Simulation Event.
    - Pedigree (source and approval)
    - Information provided by the Integrator
- Artefacts
    - Descriptions of the outputs from the composition
    - Enumerated list of the outputs from each M&S Service within the Composition
    - Composition level data collected. For example, data loggers capturing the information published over the M&S protocol (e.g. DIS) in order to allow playback at a future time.
    - Post-processing of data at the SoS level
    - Information provided by the Integrator


UC-1-1 Browse Simulations
Simulation Operator
MSaaS Portal Applications
Repository Services
ref
UC-1: Use Simulation
Trigger:Simulation Event scheduled
Review existing Compositions for relevance/applicability to use or modification for Simulation Event
ref
UC-3-3: Develop Composition
UC-3-4: Upload Composition
loop
[foreach inquiry by Simulation Operator]
Composition Query
Composition Details
Composition details reviewed include: functional capabilities, scenarios, M&S Services, M&S Resource Metadata (source/pedigree and applicability), and deployments
Select Composition
Select Deployment
Provide Simulation Execution Inputs as Appropriate for Simulation Event
Inputs include parameters/configurations for Services, scenarios, and deployments.
ref
UC-1-2: Start Composition

UC-1-2: Start Simulation

Once the Simulation Operator is ready to execute a simulation, the operator should be able to provide information about the simulation execution and provide input on how the composition should be deployed, accessed, and executed. The simulation execution should be repeatable so all the information and choices that the Simulation Operator provides needs to be captured. The choices to influence the execution provided by the Simulation Operator include the following details:

- Date/Time for the execution
    - Start the execution immediately, or schedule the execution to occur at a later time.
    - *Assumption: M&S Services, M&S Applications, and any connected systems require initialization steps, either before these are started or once they are started. These steps include loading configuration data and scenario descriptions, connecting to middleware services (e.g. HLA RTI), or awaiting action from a message or human involvement to proceed with the simulation execution. The MSaaS Implementation is responsible for deploying and executing the services and applications, but the orchestration and initiation may involve user actions if necessary.*
- Iterations – the number of times that the user would like to execute the simulation. This is only applicable for fully constructive, and fully automated events.
- Description – a textual description of the simulation event using the composition or deployment
- Access control – which group(s) and/or Simulation Operator(s) should have access to execute the same simulation event and be allowed access to the artefacts.
- Deployment options – The options within this selection should be based on the required and available computing resources including server availability in the cloud, networking capability where required for things like streaming of a user interface or connectivity to locally running systems.
- Note: A future capability may include Custom Properties – any configurable parameters that would be used to configure a service implementation that the Integrator has exposed for the Simulation Operator to have access to change. *This may beyond the scope of MSG-164 at this time.*
    - These could be pull-downs with enumerated list of options, a number range, a data file, or other machine-readable inputs.

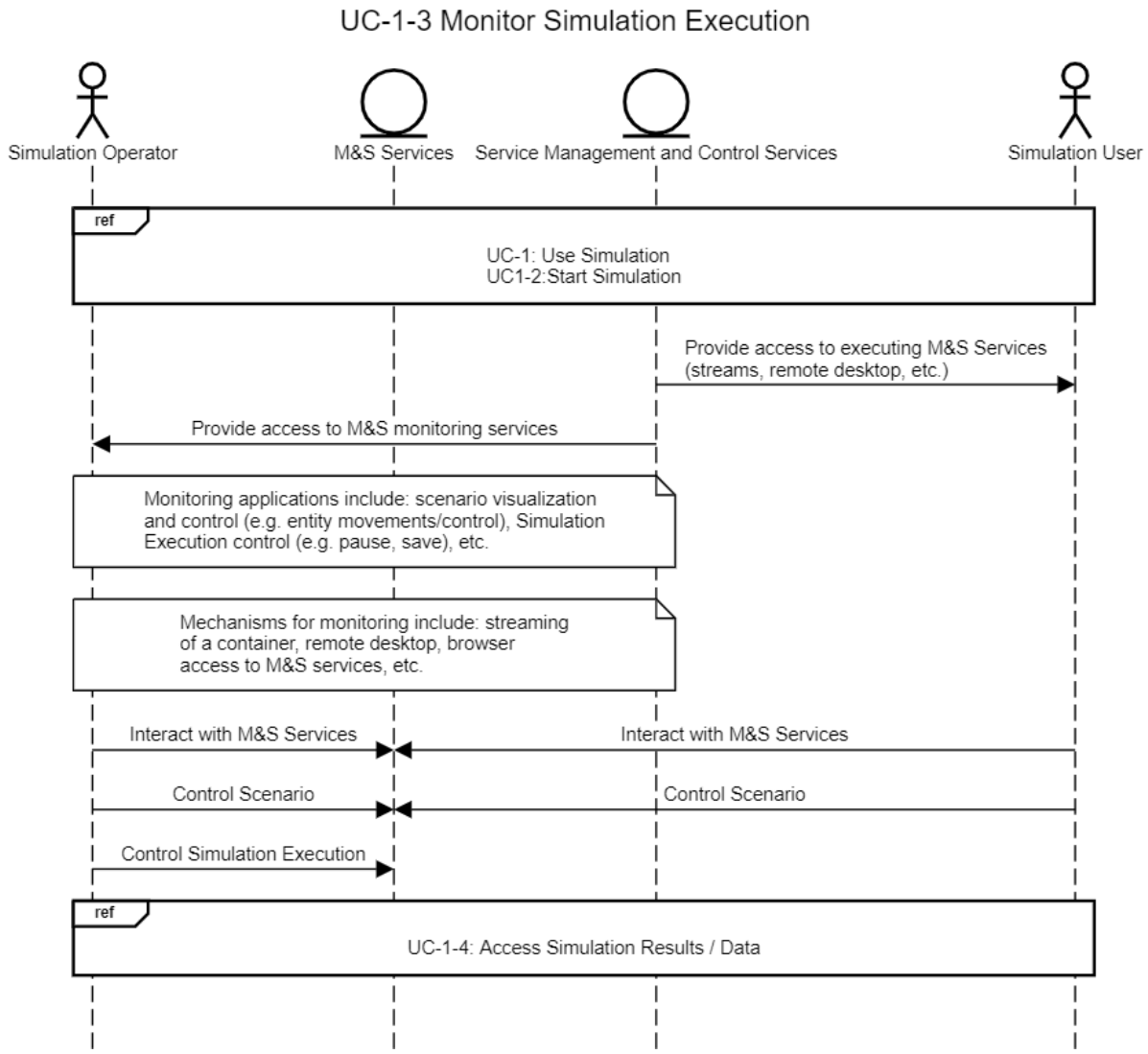


UC-1-3: Monitor Simulation Execution

When the simulation is executing, the Simulation Operator may want to view the simulation in several different ways including:

- Streaming of the output of a container or VM running in the cloud such as console output or streaming a video of the user interface of the M&S Service.
- See data displayed via the M&S Enabling Services such as a display showing the spatial location of all entities, health and monitoring information of both the scenario as well as the information technology assets (e.g. processor load or memory used).
- Connect a visualization application to the simulation environment. This requires compatibility assurances that the Integrator may have to be involved in setting up.
- The mechanisms for the Simulation Operator to interact with the executing simulation are provided by the Integrator and chosen by the Simulation Operator upon deployment selection.

The first two options would be easier to automate and would require the requisite technologies and options for the technical actors to provide, for example, installing Apache Guacamole servers on the appropriate machines / containers or providing a web page that accesses and displays run-time information via the simulation object model.



UC-1-4: Access Simulation Results / Data

Once the simulation event has concluded, the Simulation Operator will need to access the results of the simulation. The results can be varied depending on the M&S Services while also provide some typical SoS information including:

Simulation data:

- Raw data files
- Processed output files
- Imagery
- Name/value pairs in an TBD MSaaS standardized format to allow for portal displayed outcomes

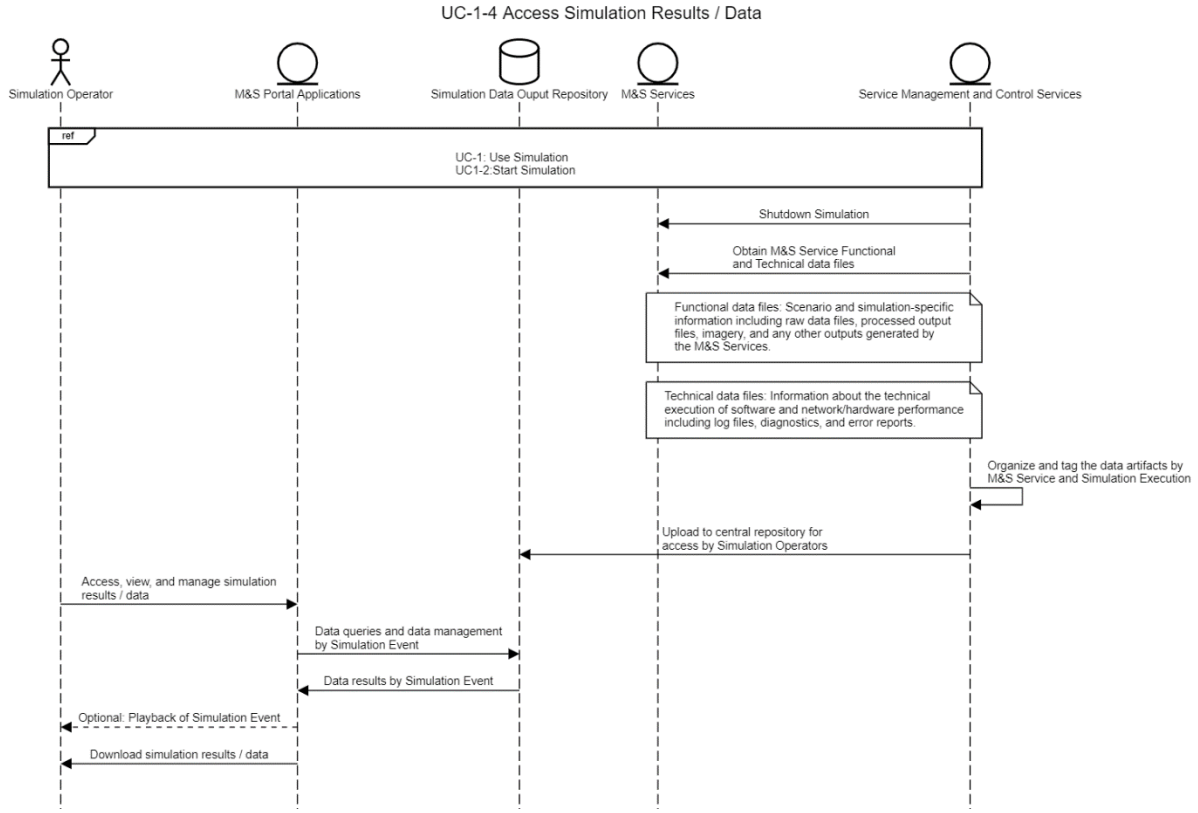
SMC data:

- Computing performance logs (e.g. CPU and memory utilization) in order to easily see if there may have been a problem

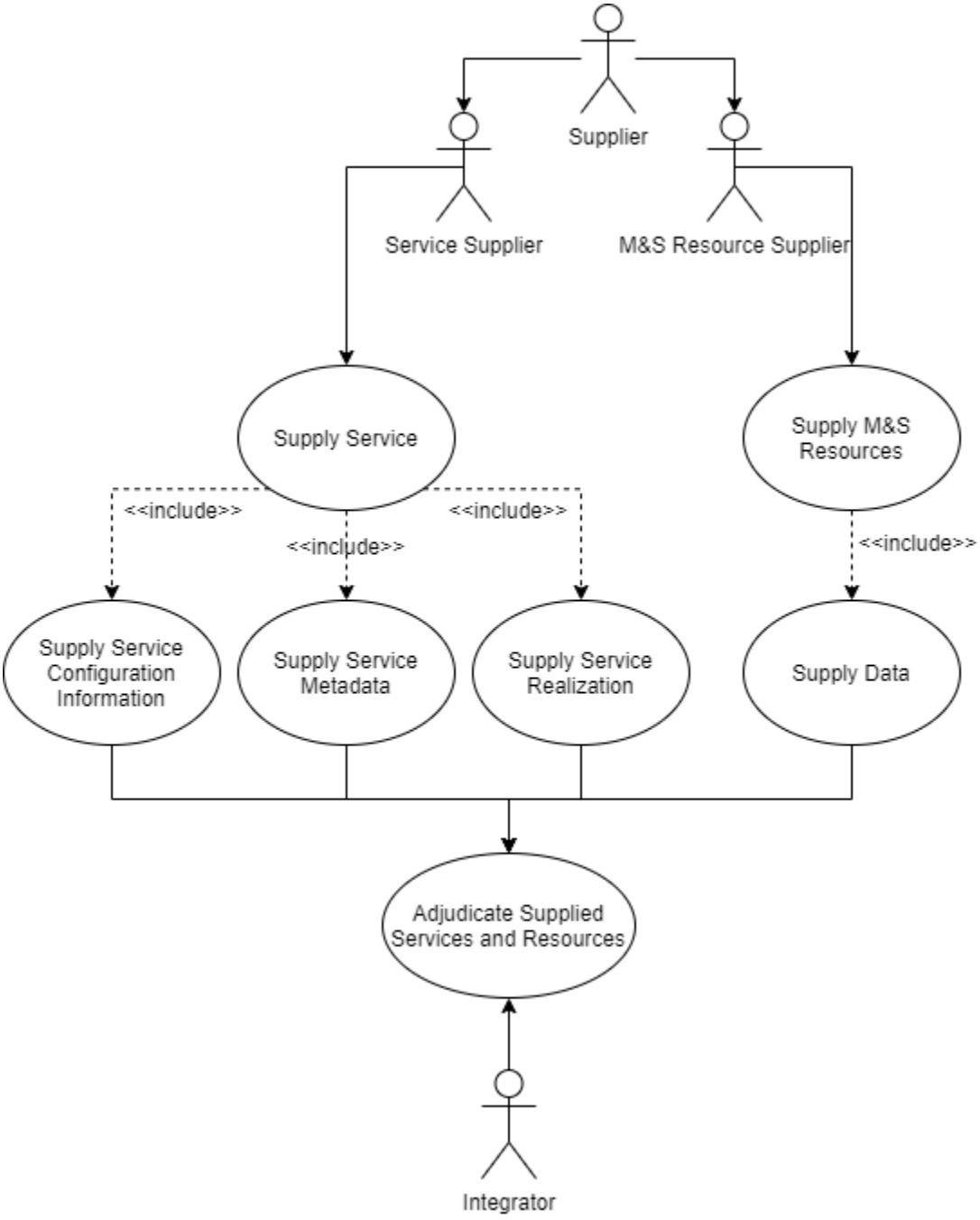
- Networking performance to determine if the network was too saturated for ideal performance or if there were any connection problems during the event
- Error logs
- Simulation management application logs

Scenario data:

- Scenario-based outputs
    - Battle-related items such as kills, casualties, shots fired, phase line times, detections, etc.
    - Movement of entities over time
- Playback – ability to view the execution at any point in time, at variable rates of playback times, pausing, etc.



Supplier Use Cases



UC-2: Supply M&S Resource

Objective: to supply M&S Resources, which includes service configuration information, service meta data, and/or M&S Service implementations, to the MSaaS Implementation and make these available to the users.

UC Actor: Supplier

UC Triggers:

- Supplier: A new version of the M&S Resource and an update of the related M&S Resource metadata
- Integrator: Request for M&S Resources (as discovered in the registry)
- Integrator: Request for additional information about an M&S Resource

Pre-Conditions:

- M&S Resources are provided as per the exercise (DSEEP) agreements, and/or standing environment policies for the MSaaS implementation (including applicable security requirements, certification and acceptance requirements, data assurance requirements, standards)
- An M&S Resource metadata template is available

Post-conditions:

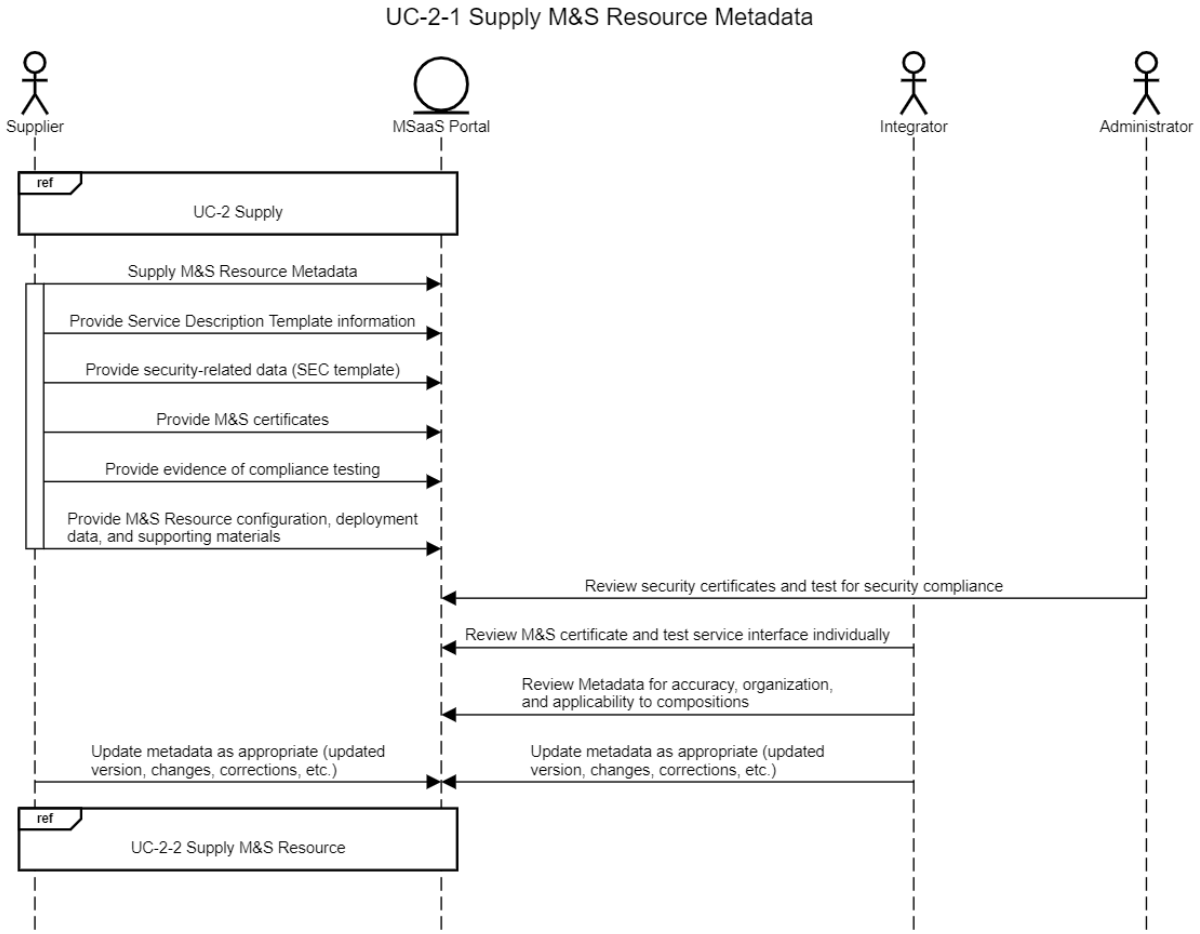
- M&S Resource complies with agreements and policies (by inspection and/or test), and available in the MSaaS Implementation
- M&S Resource metadata adjudicated (reviewed and approved, by inspection and/or test) and available in the MSaaS Implementation

UC-2-1 Supply M&S Resource Metadata

The Supplier contributes the M&S Resource Metadata to describe the M&S Resource. The metadata provides the resource's core descriptive information, functionality, domain, resolution, accessibility, interface, dependencies, and data artifacts.

Tasks:

- Provide the data in accordance with the SDT template (incl classification level of the M&S Resource, or of the data provided by the service)
- Provide security-related data in accordance with the SEC template
- Use tags/labels from a provided set or ontology and attach these to the M&S Resource
- Update metadata of previously uploaded M&S Resources
- Provide M&S certificates as evidence of certification
- M&S Resource configuration and deployment data, and supporting materials



MSG-136/164 has developed a draft Service Description Template to capture some of the information items required to describe a service. The current set of fields is included below:

Field Name	Explanation	Values
Core Information		
Service Name*	The name of the Service. Should fit the operations of the service.	Free text
Description*	A detailed description of the service, what it is used for and what it does. This description should be written in a user friendly way so that a person new to the service understands the purpose, scope and limitations of the service.	Free text
Points of Contact (PoC)*	A list of Points of Contact. This list should contain all important PoC's responsible for the service, so if there is a problem, the user knows who to contact about it.	
Status	The current status of the service. One of the following options needs to be selected at all times • Prototype - ... • Production - Service is available for use by intended customers • Retired - Service is disposed and can no longer be used	Select exactly one of: Prototype, Production, Retired

Version*	The version number of the current version of the Service. Every new version of the service will result in a new version of the description. The version number will be provided in plain text. Its format will be defined by the owner of the service.	Free text
Release Date*	Date when last version was released	Date (YYYY-MM-DD)
Pictures	Please provide screenshots, images, etc. illustrating the service.	Image
Videos	Please provide videos (if applicable) of the service, or provide reference to web location (YouTube, etc.)	Video
Additional Information (optional fields)		
Simulation Domain	Specify simulation domain	Live, Virtual, Constructive, C2
Warfighting Domain	Specify warfighting domain	Land, Air, Sea, Space, Joint
Application Domain	Specify application domain	Training, Analysis, Acquisition
Model Resolution	Specify modeling resolution of service	Entity Level, Aggregate Level, NA
Keywords	Specify keywords for the service, e.g. C3 Taxonomy category	Free text
Service Accessibility		
Website / URL*	Please provide a website where more information can be found or the repository location (e.g., URL) where the service can be obtained (e.g. where Docker image is stored)	URL
Deployment Method*	Please specify deployment method.	Select all that apply: Container, VM, other
Additional Deployment Information	Any additional information regarding service deployment, e.g. requirements or constraints associated with deployment methods	Free text, Attachments
License type	E.g. Perpetual, Subscription (per time, per use)	Select all that apply: Pay-Per-Use, Subscription, License, Free Of Charge
Additional License Information	Any additional information regarding licensing, billing (ways to pay for the service and pricing), etc.	Free text
Usage Constraints	Limitations (e.g., number of licenses, application areas) and constraints that limit the use of the service. Hosting constraints.	Free text
Service Interface		
Service interfaces *	List supported service interfaces	Free text
Additional information	Any additional information on the service interface	Free text
Service Dependencies		

Supported data exchange interfaces	List supported interfaces	HLA1516, HLAevolved, HLA4, DIS, HTTP, RPR2, NETN, DISv7, DISv8, WFS, WMS, other
Additional Information	Provide additional information regarding the supported interface standards (e.g., exact version, required data).	Free text
Security Software Assessment		
Usage Description / Capabilities	COTS, GOTS, Open Source, Other Technology stack Patch management PII processing Service information IPv6 compliance	Form including some free text
Network Analysis	Ports, protocols, inputs, and outputs	Text
Security Scans	Scan type, tool, location, and date	Text
Cybersecurity Concurrence	Assessment conclusions with POCs	Text

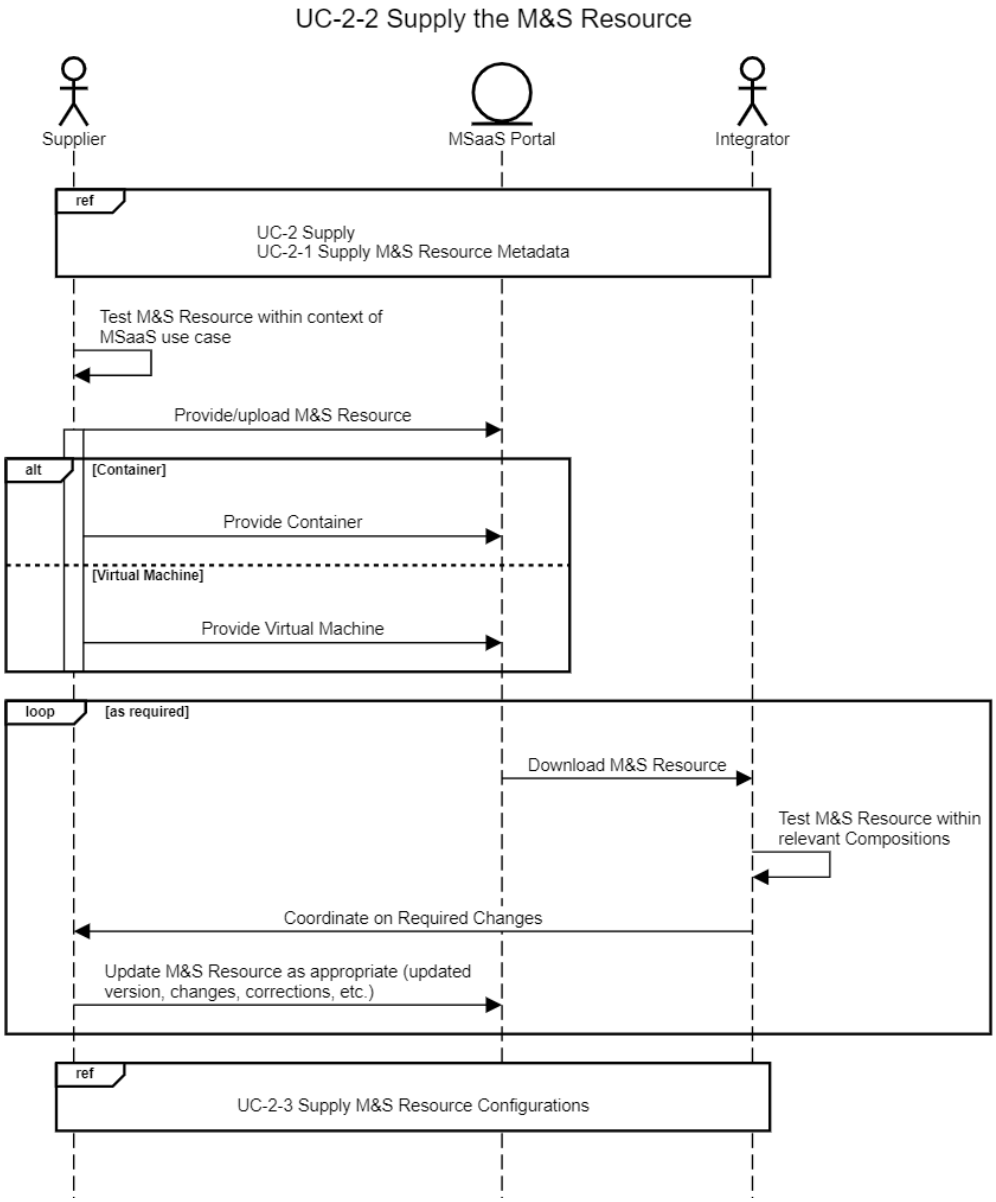
UC-2-2 Supply the M&S Resource

The Supplier supplies the M&S Resource to the MSaaS Implementation / M&S Repository Services. The Supplier can provide the M&S Resource via multiple methods. The Supplier can supply a container, virtual machine, terrain data, or the Supplier can supply information necessary to access the service that is executing elsewhere. This access to an external service would be represented by an endpoint and an interface specification for its usage.

M&S Resources may reside in Repository Services that are accessible by the Integrator in order to test and integrate, and are accessible by the Service Management and Control applications.

Tasks:

- Perform tests to verify completeness of the provided M&S resource
- Provide evidence of compliance from testing
- Upload the M&S Resource into the MSaaS Implementation (M&S Repository Services)



UC-2-3 Supply M&S Resource Configurations

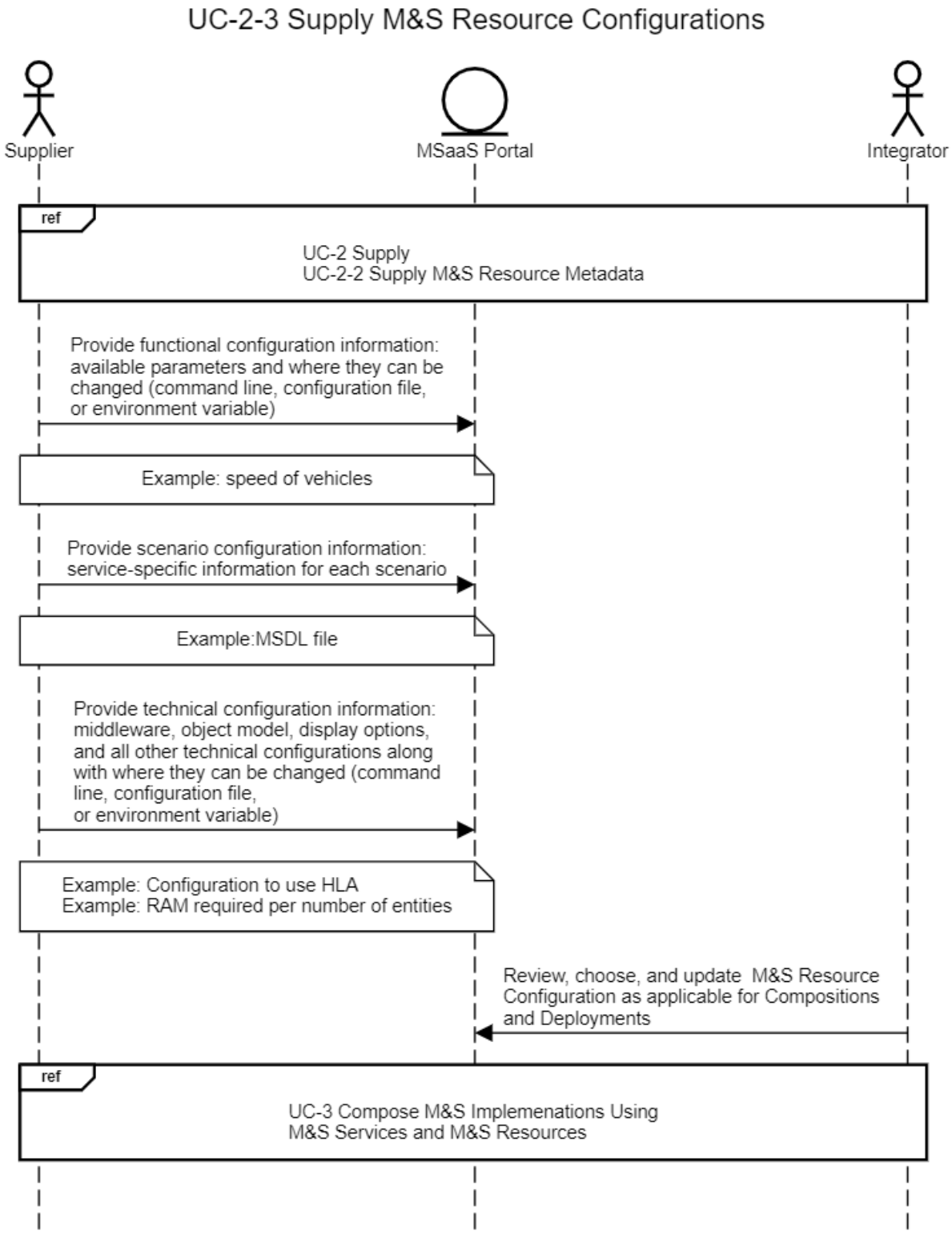
The Supplier supplies information on how to configure the service, whether in an automation method or in a manual method. Textual descriptions for how to configure the service may be in the form of typical software documentation. Data-oriented configuration information can be provided in the requisite data formats. For example, scenario configurations may be done by updating a Military Scenario Definition Language (MSDL) file. The data files that can be modified to configure the service must be identified along with any guidance for the Integrator including what can be modified, any restrictions on data fields, loading requirements, and anything else the Integrator needs to understand to effectively use and configure the service.

Configuration data is specific to the M&S Resource, the type of modeling functionality, the scenario format, and the data requirements (e.g. terrain, performance data, etc.). Therefore, configuration data

files will be stored in the M&S Repository Services and the instructions for the Integrator for modifying those files will be captured in the same place, but in human readable text format for an integration engineer to utilize. When an MSaaS Implementation has well-known and structured M&S Services for which configuration can be automated (by the Service Management and Control applications / services), the M&S Repository Services will include that specific information for automation.

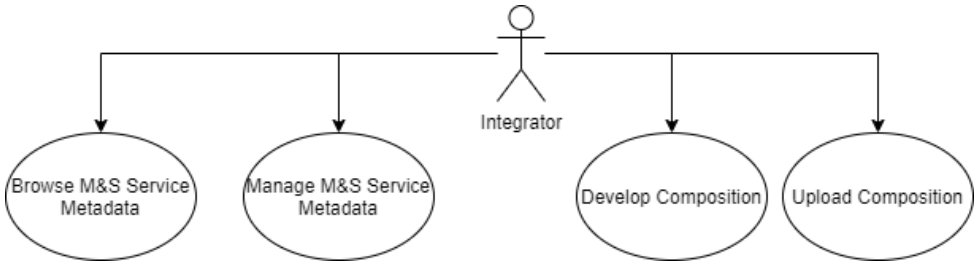
Tasks:

- Provide information how to execute a Service Implementation
- Define the minimum resources required for the M&S Services
- Define the configuration options for a Service implementation


UC-2-3 Supply M&S Resource Configurations
Supplier
MSaaS Portal
Integrator
ref
UC-2 Supply
UC-2-2 Supply M&S Resource Metadata
Provide functional configuration information: available parameters and where they can be changed (command line, configuration file, or environment variable)
Example: speed of vehicles
Provide scenario configuration information: service-specific information for each scenario
Example:MSDL file
Provide technical configuration information: middleware, object model, display options, and all other technical configurations along with where they can be changed (command line, configuration file, or environment variable)
Example: Configuration to use HLA
Example: RAM required per number of entities
Review, choose, and update M&S Resource Configuration as applicable for Compositions and Deployments
ref
UC-3 Compose M&S Implemenations Using M&S Services and M&S Resources

Below is a sequence diagram depicting the information and workflow between the Supplier and other actors in order to accomplish the development of Compositions.

Integrator use cases



IC-3: Create Compositions using M&S Services

Objective: to coordinate with Suppliers and Simulation Operators to compose M&S Resources and M&S Services into possible M&S Implementations to be used by Simulation Operators. This includes managing software, data, and metadata about M&S Services. This use case incorporates the other use cases within this section.

UC Actor: Integrator

UC Triggers:

- Integrator: Received new M&S Resources or M&S Services
- Integrator: Received request from an Operator to compose an M&S Implementation based on a Simulation Event requirement

Pre-Conditions:

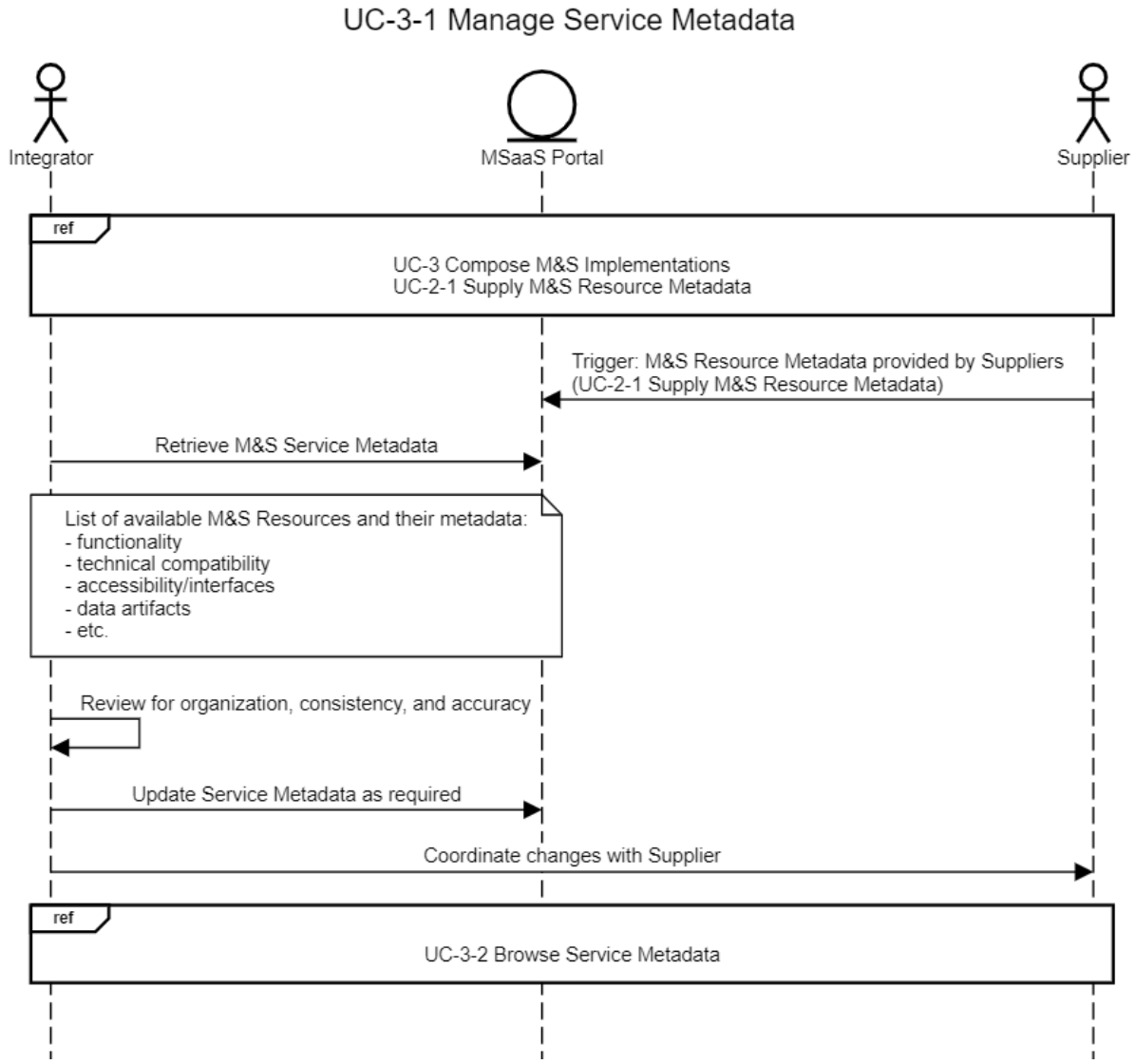
- M&S Services and M&S Resources have been made available via the M&S Portal Applications
- Simulation Event exists that requires specific composition of M&S Services based on its scenario.

Post-conditions:

- A Composition has been created and Simulation Operators have access and permission to deploy/execute the Composition via an M&S Portal.

UC-3-1: Manage Service Metadata

Given the requirements of a composition, the Integrator needs to be able to determine the applicability of each M&S service for inclusion in the composition. The information provided for an M&S service might indicate it being suitable but may be missing detail, completeness, or compliance with expected norms/taxonomies. In such cases, the Integrator will work with the Suppliers to update/revise the service metadata.



UC-3-2: Browse Service Metadata

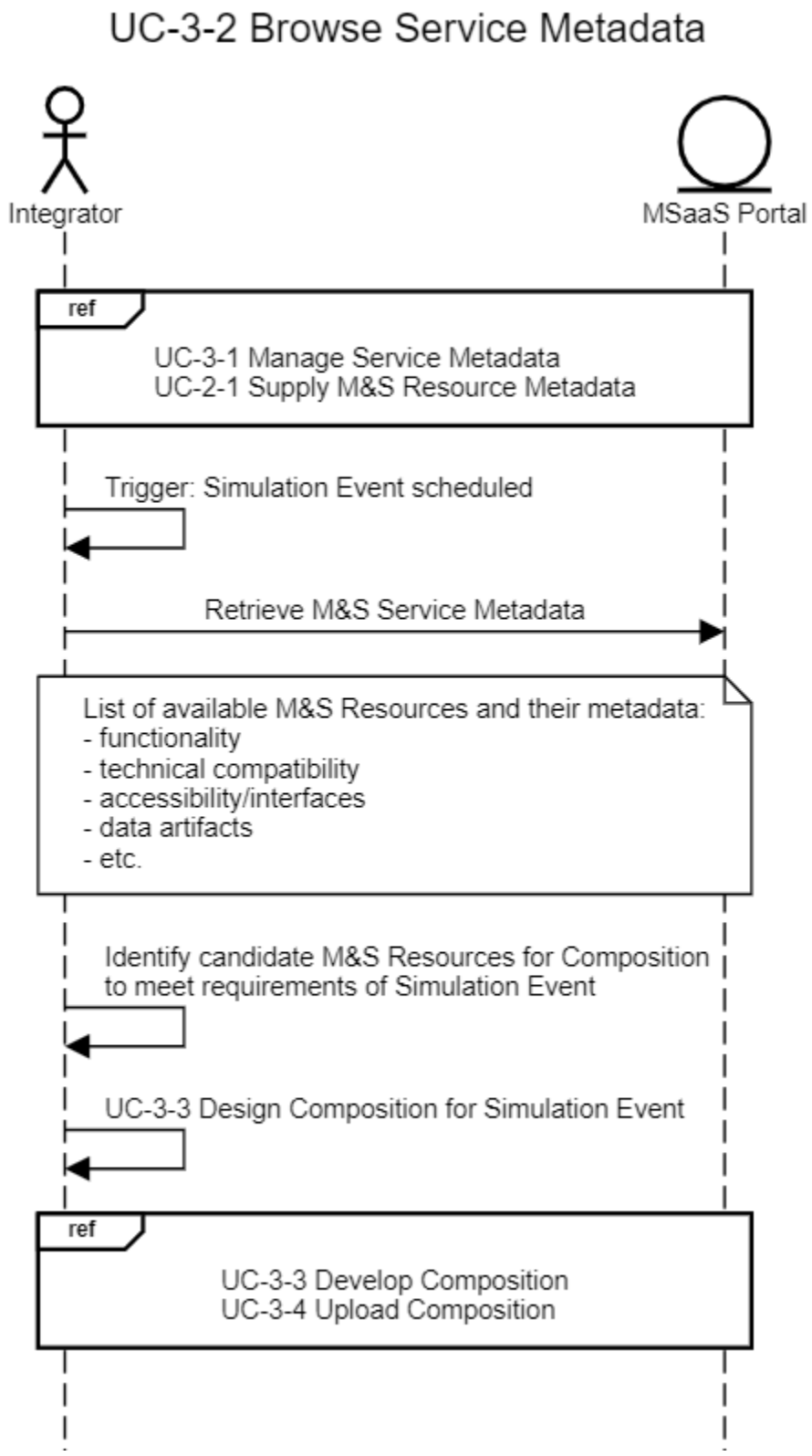
The Integrator's task is to build a composition that may be used by the Simulation Operator and to define the options that will be available to the Simulation Operator. In order to achieve this task, the Integrator is provided with simulation requirements (output of DSEEP Step 2) and they have the responsibility to identify the services to be included in the composition design to meet the requirements of a specific event.

The Integrator identifies candidate services that best support the simulation requirements by accessing the M&S Registry Services. In addition to finding services that meet simulation technical requirements, licensing restrictions, export controls, etc. need to be considered and recorded.

In particular, the retrieved M&S Resource metadata includes:

- The list of M&S Services available from the own MSaaS Implementation or from another (remote) MSaaS Implementation.

- The metadata that describes the services. The information includes the services' core descriptive information, functionality, domain, resolution, accessibility, interface, dependencies, and data artifacts, as specified in the Service Description Template (see Provide Service Metadata Use Case, Section 4.1.2.1).


UC-3-2 Browse Service Metadata
Integrator
MSaaS Portal
ref
UC-3-1 Manage Service Metadata
UC-2-1 Supply M&S Resource Metadata
Trigger: Simulation Event scheduled
Retrieve M&S Service Metadata
List of available M&S Resources and their metadata:
- functionality
- technical compatibility
- accessibility/interfaces
- data artifacts
- etc.
Identify candidate M&S Resources for Composition
to meet requirements of Simulation Event
UC-3-3 Design Composition for Simulation Event
ref
UC-3-3 Develop Composition
UC-3-4 Upload Composition

UC-3-3: Develop, Test, and Upload Composition

Each service may contain configuration details that describe how to modify the service's behavior. The Integrator shall retrieve service configuration details from the Repository Services indicated by the metadata obtained during Browse Service Metadata.

The Integrator must determine the needed configuration as a trade-off between the simulation requirements and the service configuration options (e.g., restrictions on data fields or allowed data format, etc.).

Therefore, the Integrator needs to collect the following information:

- Simulation requirements related data (e.g., MSDL, terrain description, performance, etc.).
- Service Interface for each service (from the Service Metadata).
- Configuration data file for each Service.

The output of this step will be the description of the specified composition in terms of Configured Services.

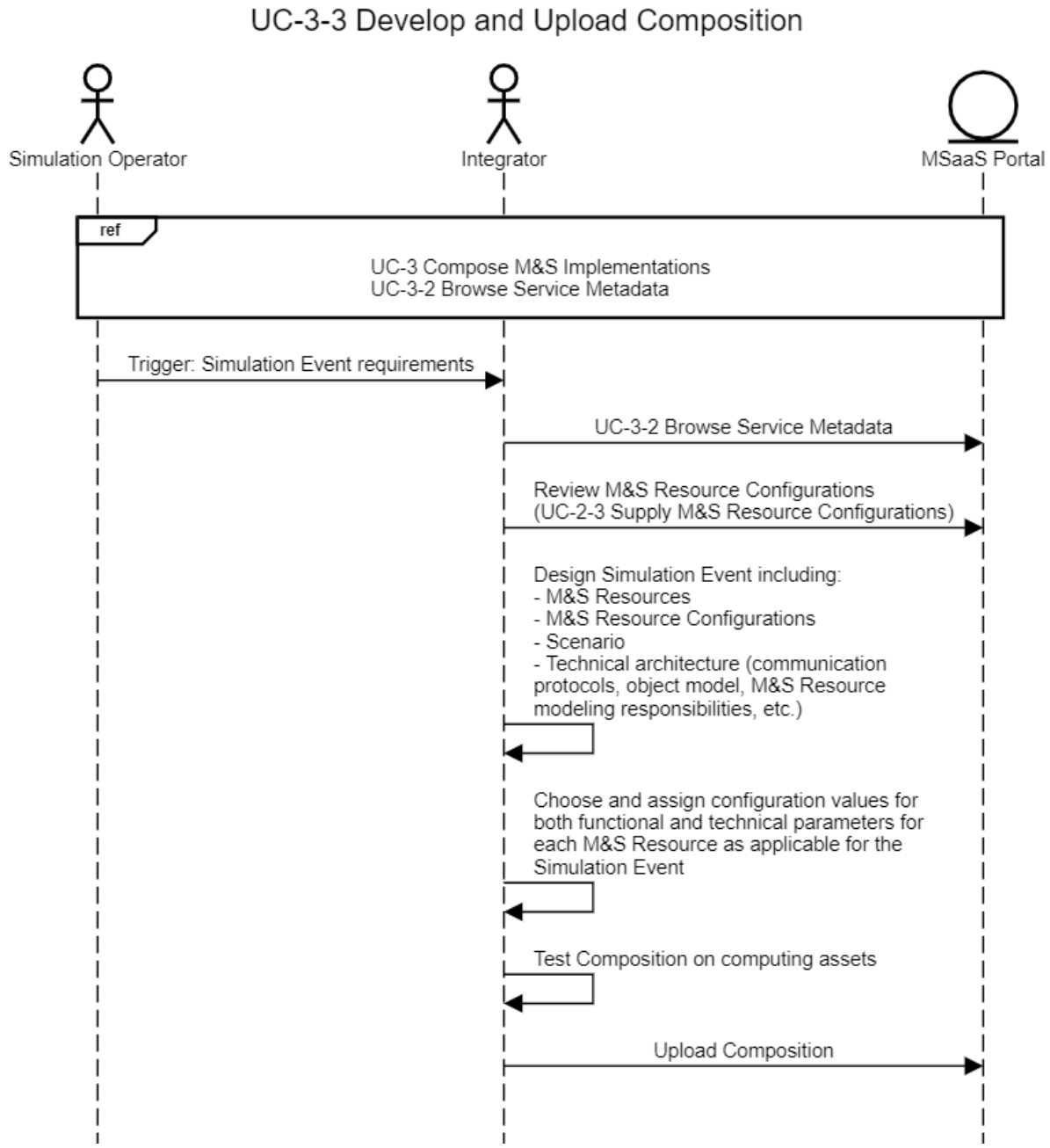
Once the required services have been configured, the next steps are to define an orchestration template consisting of:

- Computing assets (computation, networking, and other infrastructure-related resources that are needed).
- Composition of services across computing assets.

The output of this work will be:

- An orchestration template that specifies the deployment description in terms of its deployable stack
- Composition-related metadata.

The orchestration template will define compute, storage, network and other infrastructure related resources. This includes the interfaces(?) between infrastructure resources as for example binding a compute resource to a specific network.



The MSaaS Portal Applications provide the ability to upload, store, and export a composition.

How the ability to upload, store, and export a composition will be implemented is out of the scope of the Information Model. No interoperability between compositions stored in different MSaaS Implementations is considered, as it strongly depends on the underlying technology.

The Integrator stores the composition, according to the specific SMC capability, in terms of:

- Composition (i.e., list of configured services)

- Deployment description:
  - Assets (compute node, network and other infrastructure related resources that are needed)
  - How those assets are related to each other (e.g. connect compute nodes to network, etc.)

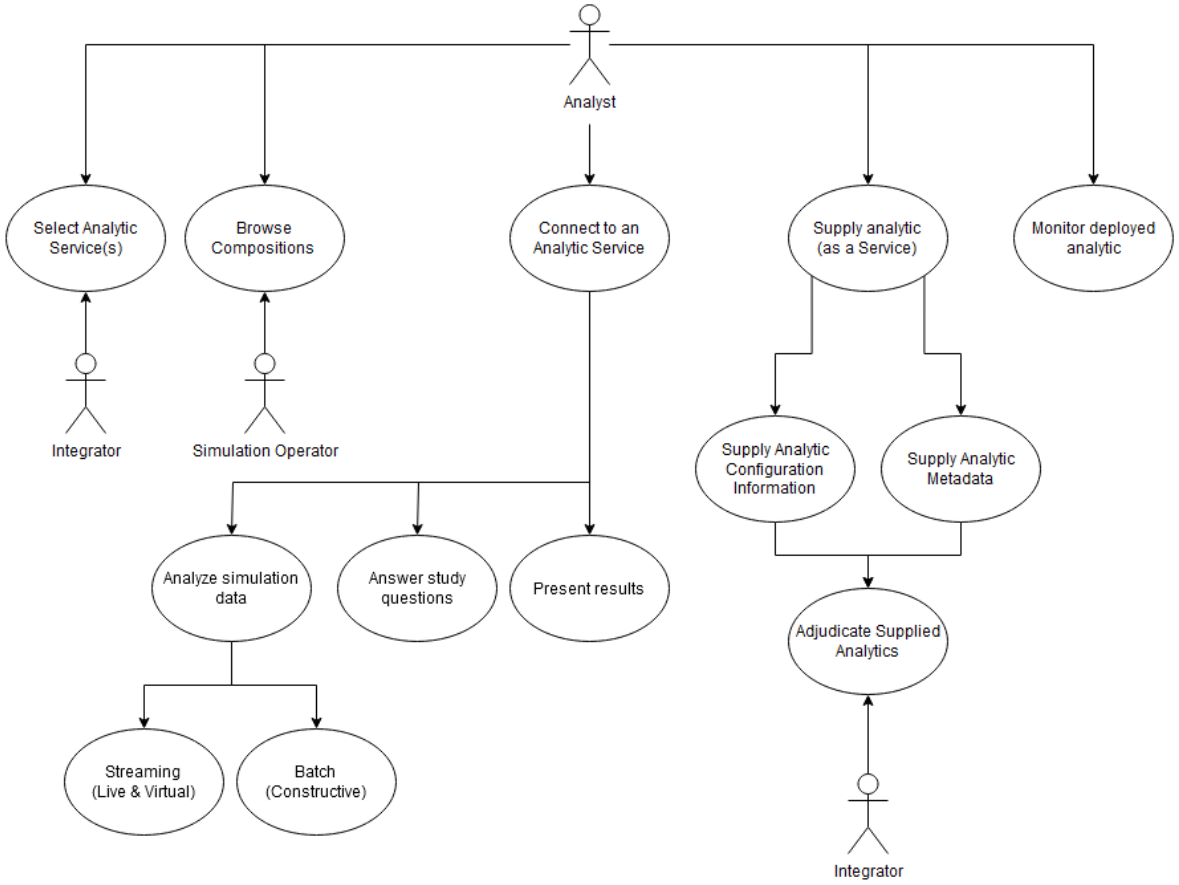
Note: It should be underlined that a widely accepted agreement on a standard or a technology for specifying compositions (i.e., orchestration templates) is still missing. As an example, the most relevant commercial players in the Cloud market (e.g., Microsoft, Amazon and Google) use different notations for their composition descriptions.

TOSCA is the OASIS attempt to standardize a vendor-neutral and YAML-based language for describing service orchestration is a promising approach. Even though the use of TOSCA could be evaluated again in the future, currently its implementation doesn't appear to be mature enough and its adoption is still too limited to be considered a valuable option.

Name	Description	Links
Get Simulation Event Details	The Integrator creates compositions that meet the requirements of a particular simulation event. These simulation event details shall be available through the MSaaS Implementation.	Use case for upload Simulation Event?
Search Service Metadata	Given the Simulation Event Details, the Integrator searches for Services that can fulfil the Simulation Event requirements. The ability to fulfil the requirements shall be ascertained through the metadata of a Service stored in a Registry.	Supplier Use Cases
Request Change to Service Metadata	When searching for services, the Integrator may find that the metadata for a Service is deficient in some way. This Use Case allows the Integrator to mark the problems with the metadata and request the relevant Supplier to make changes.	Search Service Metadata
Get Service Configuration Details	An identified Service may need to be configured to support the	Search Service Metadata

	requirements of a Simulation Event. The information for how a Service is configured is stored in a Repository along with the Service Realisation. The Integrator shall access these configuration details to understand how to create a Configured Service.	
Upload Configured Service	Once a Configured Service has been created, the Integrator shall upload it to a Repository and note its URI.	Get Service Configuration Details
Upload Composition	A set of Configured Services are brought together to create a Composition. The Composition fulfils the requirements of the Simulation Event.	Upload Configured Service
Attach Deployment to Composition	A Composition can be deployed to different Simulation Infrastructures. This deployment information is attached to a Composition.	Upload Composition

Analyst use cases[5]



UC-4: Analyze Simulation Results

Objective: to execute a simulation with analytic objectives, which includes working with Integrators and Simulation Operators to browse existing compositions and deployment descriptions and to create new compositions; accessing and analyzing simulation data created within these compositions and monitoring the performance of deployed analytics within a composition. The other use cases in this section are included within this use case.

UC Actor: Analyst

UC Triggers:

- *Operator: A simulation event exists, with analytic objectives, and requires a composition of M&S Services to be executed at the event. This simulation event may be live or virtual (human in the loop, in real time) or constructive (may/may not have human in the loop, likely faster than real time).*

[5] Text with italics in this section is new material contributed by Neil Kester. Un-italicized text comes from existing NATO MSaaS work.

Pre-Conditions:

- *Analytic Resources – including composition and deployment descriptions – are provided by the M&S Repository Services. These elements describe how the M&S Service may pass data with the Analytic Resource and how the Analytic Resource is deployed. (These are provided by the "Supply Analytic Services" use case.*
- *Analytic Resource Metadata is provided by the M&S Registry Services. (This is provided when the Analytic Service is registered through the "Supply Analytic Services" use case.*
- *The MSaaS Portal Applications provide access to the Analytic Metadata. Ideally, this is through the same interface as M&S Metadata where the services are differentiated by existing Metadata tags.*

Post-Conditions:

- *Composition and deployment descriptions are created for the simulation event.*
- *Specific simulation event-related configurations of the deployment's analytic environment are created. This includes architectural building blocks (ABB) types, architectural patterns (APs) used, and resources allocated during the event.*
- *The simulation and analysis are executed. Any analytics created are submitted back into the registry for inclusion as a "Deployed Analytic Service" available to future uses of this composition. This action is executed through the "Supply Analytic Services" use case.*

UC-4-1: Select Analytic Service(s)

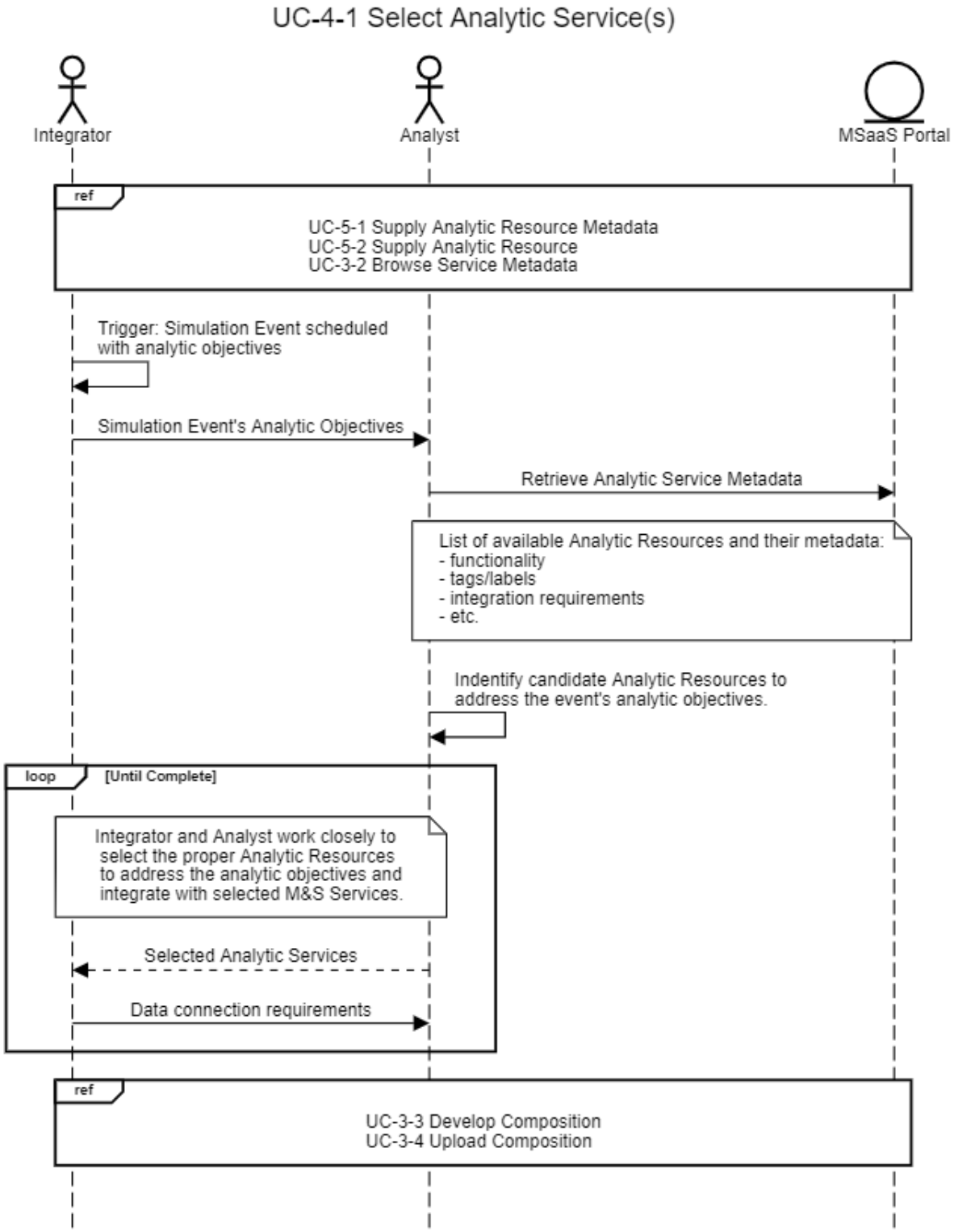
The Analyst conducts this use case in conjunction with the Integrator's Browse Service Metadata (UC-3-2) and Develop and Upload Composition (UC-3-3) use cases. As the Integrator builds a composition for the Simulation Operator, the Analyst selects from available Analytic Services those required to meet analytic objectives of a specific event. These objectives should be documented within the simulation requirements document the Integrator uses to build the Composition.

To select the proper analytic services, the Analyst (or Analytic Team) must translate analytic objectives described in the specific event's requirements document into an analysis plan. This requires decomposing those analytic objectives into analytic questions, data sources that could answer those questions, and methods or approaches to use on those data. For this reason, it is critical that the Analyst work closely with the Integrator. Both must ensure the selected M&S Services and their integration produce the required data.

The Analyst works with the Integrator to select appropriate Analytic Services and to configure and integrate M&S Resources with the analytic questions in mind. Simple configuration changes and data processing rules within or between the M&S Resources may enable the event's ability to answer those analytic questions.

In addition to data traveling from the M&S Resources to Analytic Services, Analytic Services may send data, in the form of configuration instructions to M&S Resources to configure and initiate additional simulation runs. This is especially useful in the case of executing large design of experiments in support of acquisition decisions. This feature is beyond the scope of the current research.

In the case that required Analytic Resources do not exist, the Analyst may request those resources be developed for future use.



UC-4-2: Browse Compositions(s)

The Analyst conducts this use case in conjunction with the Simulation Operator's Browse Compositions (UC-1-1) use case. The Simulation Operator and Analyst use a user interface to explore and/or search for composition and deployment descriptions to execute for the simulation event.

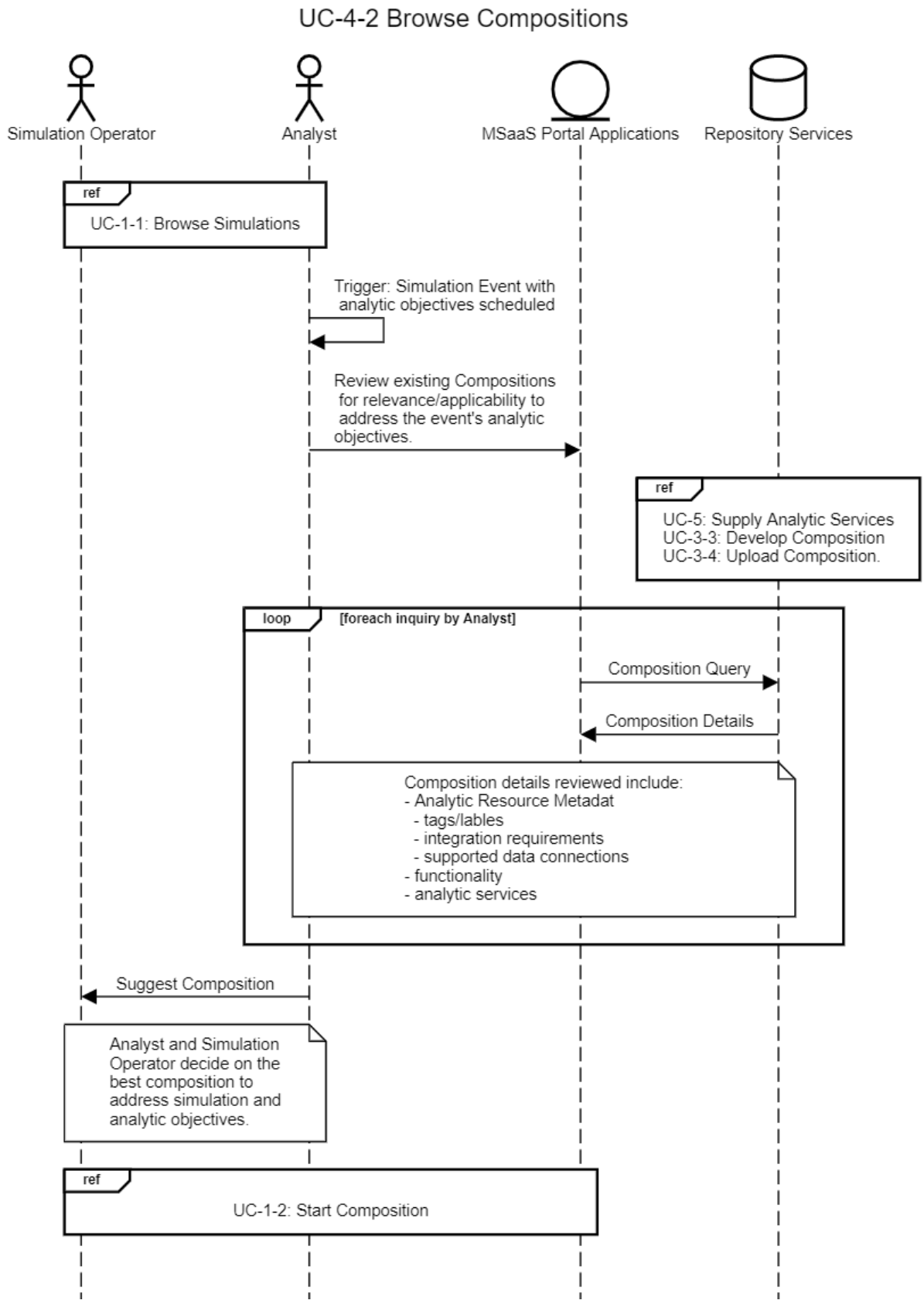
As the Simulation Operator is viewing information about compositions and deployments and their linked M&S Resources, the Analyst is viewing information about their linked Analytic Resources and the data sources available to those Analytic Resources.

Just as the Simulation Operator evaluates a composition or deployment's scenario description for applicability to a specific event's simulation objectives, so does the Analyst evaluate these descriptions for applicability to a specific event's analytic objectives. The Analyst needs the following composition details for that evaluation (to be included in the composition description):

- Functional capabilities of the composition describing what is represented in the simulation, and to which level of detail:
    - These capabilities could be simply keywords (e.g. medical supplies) or they could be more detailed and based on a central (domain) taxonomy (or an ontology) that enforces common vocabulary and denotes relationships among functions.
    - Capabilities ~~could~~ *should* include:
        - a link to specific term/meaning within the functionality taxonomy/ontology
        - pedigree (source and approval) of the models.
        - fidelity – accuracy (to be determined how to represent this in a standard way)
        - resolution – detail (to be determined how to represent this in a standard way)
        - *Run-time speed (real time and/or faster than real time)*
        - *Human-in-the-loop or not*
        - *error – how uncertainty is represented in each model and at the composition level.*
        - Additional metadata including model limitations, assumptions
    - Information provided by the Supplier
- Related simulation scenario(s):
    - Location – place in the world that the scenario is occurring and/or if the location is fictional, then providing some geo-typical information such as a desert, forest, etc.
    - Entity information
        - Force laydown / initial positions
        - Force types (e.g. future force: a proposed future force structure)
        - Planned movements / actions
        - Entity types and count
        - Unit aggregation when possible. For example, saying that a Brigade is at a location rather than showing where every single entity is would be easier to view unless the user wants to zoom in with that level of detail
        - *Human-in-the-loop or not*
        - *Possible entity configuration parameters. For example, each entity of type "Infantryman" can be equipped with a 1 x weapon, 1 x optic, 2 x communication*

devices (e.g. radio), 1 x ruck (specify weight). This enables trade-space analysis and unit customization.

  - Information provided by scenario developers and assigned by the Integrator
- *Related Analytic Resource(s):*
  - *List of available resources (Integrated Development Environments, analytic tools, database management systems, data connections to M&S Resource inputs and outputs, deployed analytics)*
  - *Potential analytic questions to pose and answer with the given composition.*
  - *In addition to the enumerated list of outputs described in the "Artifacts" section, an enumerated list and data description of data flows developed to address the analytic questions described previously.*
  - *Configuration and/or access instructions for all Analytic Resources.*
  - *Information provided by the Analyst Supplier.*
- Dependencies on external systems and services:
  - Although it would be ideal if the user was only concerned about functionality, this can be important when the user is required to use a specific (external) system for their simulation event. *In the case of Analytic Services, this could be access to external package repositories (CRAN, PyPI, JuliaHub, etc.). In some cases (like that of a Deployed Analytic Service) those resources should be included in the compositions. In the case of a research environment, however, this is impractical as it is not possible to pre-determine all possible combination of required packages and versions. If required, these resources may be mirrored and hosted from within an MSaaS Instantiation but that requires overhead to maintain.*
  - Information provided by the Integrator *and Analyst.*
- Data
  - Description of the data used by the M&S Services to execute the Simulation Event.
  - Pedigree (source and approval)
  - Information provided by the Integrator
- Artifacts
  - Descriptions of the outputs from the composition
  - Enumerated list of the outputs from each M&S Service within the Composition
  - Composition level data collected. For example, data loggers capturing the information published over the M&S protocol (e.g. DIS) in order to allow playback at a future time.
  - Post-processing of data at the SoS level
  - Information provided by the Integrator


UC-4-2 Browse Compositions
Simulation Operator
Analyst
MSaaS Portal Applications
Repository Services
ref
UC-1-1: Browse Simulations
Trigger: Simulation Event with analytic objectives scheduled
Review existing Compositions for relevance/applicability to address the event's analytic objectives.
ref
UC-5: Supply Analytic Services
UC-3-3: Develop Composition
UC-3-4: Upload Composition.
loop
[foreach inquiry by Analyst]
Composition Query
Composition Details
Composition details reviewed include:
- Analytic Resource Metadat
- tags/lables
- integration requirements
- supported data connections
- functionality
- analytic services
Suggest Composition
Analyst and Simulation Operator decide on the best composition to address simulation and analytic objectives.
ref
UC-1-2: Start Composition

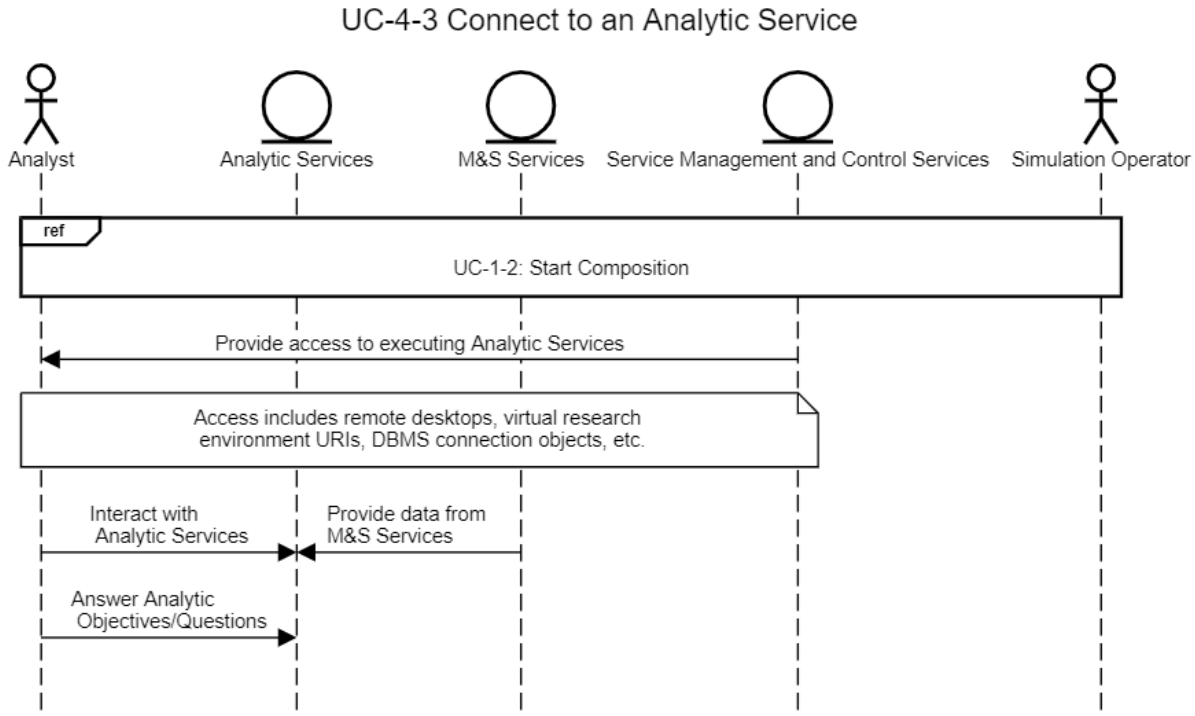
UC-4-3: Connect to an Analytic Service

When a composition or deployment is used (through UC-1-2 Start Simulation), the composition will initiate any Analytic Services included in the composition or deployment. The Analyst must then have a way of connecting to these services. This could be through a web-browser to the server based Integrated Development Environment, or via the command line via an SSH connection (to a database management system for instance).

These connection instructions should be an artefact of UC-1-2: Start Simulation as they will change from deployment to deployment. This includes connection instructions for each Analytic Resource (Integrated Development Environments (IDE), Database Management Systems (DBMS), etc.) specified in the composition.

Once these connection instructions are obtained, the Analyst is able to connect to the selected IDE using previously defined usernames and passwords. From within the IDE the Analyst can connect to a DBMS created as an Analytic Resource or a DBMS from the M&S Services (provided login credentials and access controls are in place). While data sources should have been defined and established when building the Composition in UC-4-1: Select Analytic Services, analysis may require access to other data within the M&S Services. The Analyst can do this if provided with required login credentials and network routing to those resources.

Finally, the Analytic Services should be able to connect to the simulation controller in fully constructive and fully automated events. These connection interfaces will have been defined and established when building the Composition in UC-4-1: Select Analytic Services. This allows the Analyst to create Design of Experiments (DOE) in which parameters within the simulation (initial conditions, entity definition, weather, etc.) are changed programmatically to answer specific analytic questions. This also allows the analysis of one simulation run to inform future run parameters. This feature is beyond the scope of this research but should be pursued in the future.



UC-4-4: Monitor Deployed Analytic

One type of Analytic Service is a Deployed Analytic. These are produced by Analysts for specific Compositions to answer specific Analytic Questions. They differ from Research Environment Services in that they are the finished product of an analytic workflow. This means that they ingest specific data sources, apply specific transformations and analytic methodologies to those data, and display the results in a specific way.

When a Deployed Analytic Service is included in a Composition, the Analyst must monitor it to ensure it performs as expected. This means ensuring that the data provided by the composed simulation remains appropriate for that Deployed Analytic's transformations and methods. If the Deployed Analytic employs any predictive modeling, the Analyst should monitor those models for "model drift" which describes a decrease in the model's predictive power due to changes in the input data. This may require the Analyst to retrain the model with current data.

The "consumer" of deployed analytics could be Simulation Operators, Simulation Users, Decision Makers/Leaders, and sometimes Analysts.

UC-5: Supply Analytic Services

Objective: to supply Analytic Resources for use in M&S Compositions and Deployments. This includes providing service configuration information, service metadata, and/or Analytic Service implementations to the MSaaS Implementation. These, in turn, are available to users to include in their compositions or deployments. Analytic Services come in two types: Research Environment Services and Deployed Analytic Services.

UC Actor: Analyst

UC Triggers:

- *Analyst: A new Research Environment Service is created, or a new version of an existing Research Environment Service is created. The MSaaS instantiation requires this new resource and the metadata that describe it.*
- *Analyst: A new Deployed Analytic Service is created to support an existing composition.*
- *Analyst or Integrator: Request for a new Research Environment Service or a Deployed Analytic Service.*
- *Analyst or Integrator: Request for additional information about a Research Environment Service or a Deployed Analytic Service.*

Pre-Conditions:

- *Analytic Resources are provided according to standing environment policies for MSaaS implementation (including applicable security requirements and standards).*
- *A Research Environment Service or a Deployed Analytic Service metadata template is available.*

Post-Conditions:

- *Analytic Resource complies with standing environment policies (security and standards) for MSaaS implementation. This is enforced through inspection or automated testing.*
- *Analytic Resource metadata is adjudicated (reviewed and approved, by inspection and/or test) and is available in the MSaaS Implementation.*

UC-5-1: Supply Analytic Resource Metadata

The Analyst provides the Analytic Resource Metadata to describe the provided Analytic Resource. This includes the Analytic Resource's description, functionality, interface requirements (protocols, data structures, ports, etc.), connection instructions, external dependencies, licensing requirements, and which MSaaS domains it is suitable for (training, acquisition, etc.).

Tasks:

- *Provide the data in accordance with the broader MSaaS implementation's Service Description Template (SDT). This provides information like the service name, description, points of contact, status, and accessibility information. See UC-2-1 for a complete list.*
- Use tags/labels from a provided set or ontology and attach these to the Analytic Resource
- Provide security-related data in accordance with a security template.
- Update metadata of previously uploaded Analytic Resources.
- If applicable, provide analytic model certificates as evidence the model(s) included in a Deployed Analytic have been validated.
- Provide data that describes how to configure and deploy the Analytic Resource, including supporting materials.

UC-5-2: Supply Analytic Resource

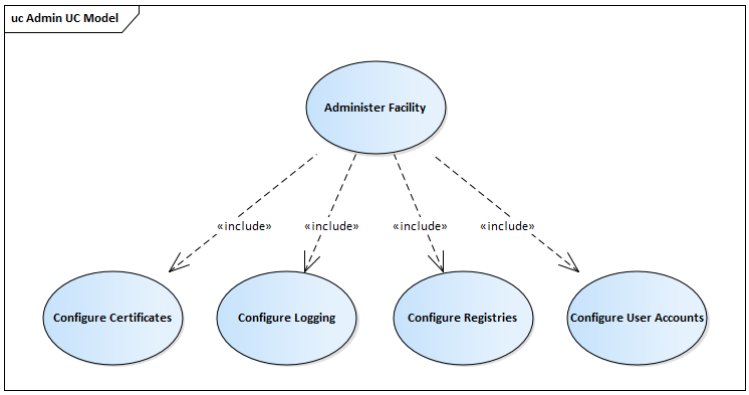
The Analyst provides an Analytic Resource to the MSaaS Implementation's Analytic Repository Service. This Analytic Repository Serveric may be a component within the MSaaS Implementation's M&S Repository Services. The Analyst should provide the Analytic Resource in a form that matches the technical capabilities of the specific MSaaS Implementation. This may mean a container or set of containers, a virtual machine or set of virtual machines, or the information required to access an external resource that will provide this service. If a container solution, the Analyst should provide the required container orchestration resource specifications packaged in a helm chart. If a virtual machine solution, the Analyst should provide the required virtual machine images or infrastructure as code (IaC) resources required to build those virtual machines (e.g. Ansible Playbooks). If an external resource, this should be represented by an endpoint and an interface specification.

Once submitted and accepted, the Analytic Resource resides in the MSaaS Implementation's Analytic Repository Service for identification and accessibility by the Integrator and other Analysts through use cases within UC-4-1 and UC-4-2.

Tasks:

- *Perform tests to verify the completeness of the provided Analytic Resource*
- *Provide artifacts certifying successful compliance testing*
- *Upload the Analytic Resource into the Analytic Repository Service provided satisfactory compliance testing results*

Administrator use cases



The Administrative tasks required for the MSaaS infrastructure include elements across a wide spectrum of professional expertise. These tasks will likely be done by a group of Administrators rather than a single user. For the purposes of this document, this group of people will be referred to as the Administrator users collectively. Each organization will need to determine the types of professionals required for their respective MSaaS environment.

The Administrator users are responsible for maintaining the computers, operating systems, registries, networks, and any other non-simulation asset. They are also responsible for the user accounts and group allocations. Finally, the Administrator users are responsible for the security of the computing infrastructure including applying security patches, ensuring operating systems and core software are up to date, logging access, and all other MSaaS infrastructure requirements levied by the nation owning the infrastructure.

Administrator users must work closely with the MSaaS infrastructure's organization to align the IT resource requirements with the organization's business strategies including collaboration with other MSaaS infrastructure implementations and their respective organizations. IT requirements should be directly derived from the required technical capabilities of the MSaaS environment. This also includes adherence to policies for utilization, security, and performance.

Manage Certificates

The Administrator users will manage all certificates required to ensure encrypted and secure communication across all computing resources. Secure Sockets Layer (SSL) or Transport Layer Security (TLS) certificates, standards, and best practices should be used for communication within the MSaaS infrastructure. The Administrators will be responsible for procuring, managing, and updating these certificates. Refer to the MSaaS Cyber Security materials for more details on certificate requirements and best practices.

Manage Logging

The Administrator users will configure all systems to have proper logging for their organization's requirements. Logging should include multiple types of information: computing performance, security, application performance, simulation execution performance, simulation scenario data, and MSaaS execution. Below is a table describing the types of information at each one of these layers:

Logging Layers	Information
Computing Performance	Computer and network performance data including virtualization. This type of information is valuable in understanding where computers, network, or virtual machines require updates, have problems, and can also be used for the MSaaS infrastructure to more optimally deploy applications and understand expected performance for the Simulation User.
Security	These logs will help monitor actions as well as provide traceability and accountability. This will allow security applications to notify administrator users of important events, see what user triggered the event, and allow for early detection and rapid correction. Critical security-related events identified in the organization's audit policy should be logged along with the authority / user that executed the event.
Application Performance	Applications within the MSaaS infrastructure should include logging to document their performance and actions taken. Applications include MSaaS infrastructure applications that support the execution of simulation as well as the simulation applications.

Simulation Execution Performance	The majority of simulation that occurs within an MSaaS includes multiple distributed applications. The performance of the overall System of Systems (SoS) environment should be logged and monitored in addition to the individual applications. Log files can be captured from a middleware application (e.g. the High Level Architecture (HLA) Run-Time Infrastructure (RTI) application), or any application that directly or indirectly supports the execution of the SoS.
Simulation Scenario Data	The data created by the simulation should be logged to not only provide the data to the Simulation users, but also to monitored unexpected effects of the computing infrastructure performance on the simulation performance. Delayed delivery of messages within the MSaaS infrastructure could cause changes in the scenario's operation. For example, if a message is dropped or delayed, a simulation could take difference actions than it would've taken if the message had been delivered as expected.
MSaaS Execution	The MSaaS infrastructure and its functionality should also be logged to detect errors, improve performance, and ensure proper execution of the simulation environment. Applications include the MSaaS portal and all of its deployment, configuration, execution, and monitoring capabilities/applications.

Log files require on-going monitoring and maintenance. Security and log monitoring applications should have access to the log files as necessary. Log files should be archived and eventually removed per the organization's security policy. Log files should also be protected by having proper permissions only allowing proper applications and users to create, view, edit, and remove the log files.

Manage Registries

The Administrator users will be responsible for the creation, and management of the software / container registries. The Administrator will manage user access and privileges for the registries, the performance of the registries, and all other IT-related management of the registries. The Administrator users will not be responsible for the information / applications within the registry, which is the responsibility of the Integrator and Supplier users. The type and implementation of registries is dependent on each MSaaS infrastructure implementation, therefore the details of the management of the registries is specific to each MSaaS infrastructure, but with all MSaaS implementations, the Administrator users are responsible for the registries availability, security, and proper operation while the Supplier and Integrator users are responsible for the contents of the registries.

Manage User Accounts & Groups

The Administrator users will create, manage, monitor, and remove user accounts as required by the MSaaS infrastructure. The user accounts will be of a specific type (Simulation, Integrator, and Supplier Users), or multiple types, which will grant them the appropriate privileges to accomplish their requisite functions. Users can additionally be assigned to groups which can allow for compartmentalizing of

information within the MSaaS. Applications and data can have limited access to certain groups allow multiple countries, organizations, or users to use the MSaaS infrastructure while retaining proper privacy based on the data or applications' policies. For example, applications that have export controls can be limited to groups that have users from the nation requiring the application to be limited.

Elements that should be assignable to groups include simulation applications, simulation data, simulation logs, and MSaaS execution details. This could include access to registries / databases or portions of the registries / databases. Access checks should be done at multiple layers including the MSaaS portal, the registries, and the operating systems of virtual and physical machines. This will require a centralized management of users, groups, access rights, and security policies.

Continuous Information Technology (IT) Management

IT management in this case refers to the procurement, installation, monitoring, and administration of the MSaaS infrastructure's hardware, operating systems, MSaaS core software, and networks. IT management of simulation applications is not the responsibility of the Administrator users, but rather the Supplier and Integrator users.

The Administrator users will manage the IT infrastructure so that it is updated according to security best practices as well as ensure that the computing infrastructure is available to users as much as possible. This includes data analysis of log files, security event notifications, availability testing, policy compliance, installation, and configuration of new assets, and many more functions as appropriate for each MSaaS infrastructure implementation.

Appendix 2: Using {modSim}

Neil Kester

5/10/2021

Step 1 Create the PostgreSQL Database

This should only be done initially.

These are the required parameters:

```
#> These source files read in the mongoURI and mongoDb elements and the Postg
reSQL connection objects.

source("./connectionObjects/pgConnectionObj.R")
source("./connectionObjects/mongoConnectionObj.R")

#> Add to the pgConnParam the database name

pgConnParam[["pgDb"]] <- "modSim"

#> Specify the desingPoints
```

This function connects to a PostgreSQL instance, creates a database (of the name provided in the previous chunk), and creates all of the tables and views shown in the entity relationship diagram (ERD).

```
modSim::Step1_createModSimDb(connParamList = pgConnParam,
                             Materialized = FALSE)
```

Step 2 Extract Transfrom and Load (ETL)

Once the PostgreSQL database is created, the simulations have run, and their results are written to your MongoDB. It is time to select data, structure it, and write it to PostgreSQL.

These are the required parameters. They are in addition to the ones provided in step 1.

```
{ # Select a design point or design points

  designPoints <- c("A1",
                    "A2",
                    "A3",
                    "A4",
                    "B1",
                    "B2",
                    "B3",
                    "B4",
                    "C1",
                    "C2",
```

```
                    "C3",
                    "C4")

} # close Select a design point or design points
```

This function executes one design point at a time. In order to execute list of design points we will execute this function in a loop. The `Materialized` argument tells the function if the PostgreSQL database you are writing to uses Materialized Views or traditional Views. Materialized Views greatly improve query performance as all data in the view has already been executed and the results are stored in another table. Traditional Views, conversely, execute those queries on demand. The downside of Materialized Views is that they consume additional disk space.

```
for(designPoint in designPoints){

  modSim::Step2_queryMongoAndFillPg(mongoConnParam = mongoConnParam,
                                    pgConnParam = pgConnParam,
                                    designPoint = designPoint,
                                    Materialized = FALSE)

}
```

Step 3 Analyze and Graph

This step queries data from PostgreSQL, conducts some minor transformations and aggregations and then graphs the resulting data. In order to isolate results, we need to provide these functions with the following parameters.

- `sensorForce`: What force does the sensor belong to?
- `targetForce`: What force do the targets belong to?
- `sensors`: What specific sensors do we care about? This should be the name used in the simulation. Also, it must be on the same force as is specified in the `sensorForce` parameter.
- `losColor`: In the graphs, what color do you want to represent line of sight (LOS).
- `acqColor`: In the graphs, what color do you want to represent sensor acquisitions (ACQ).

```
library(magrittr)

sensorForce <- "BLUEFORCE"
targetForce <- "REDFORCE"

sensors <- "WASP 1" #c("EPBV 90 1","EPBV 90 2","WASP 1")
```

```
losColor <- "blue"

acqColor <- "black"
```

Query PostgreSQL for the data

This function connects to the PostgreSQL database and queries it for the provided parameters. It saves the results as a variable called `graphData`.

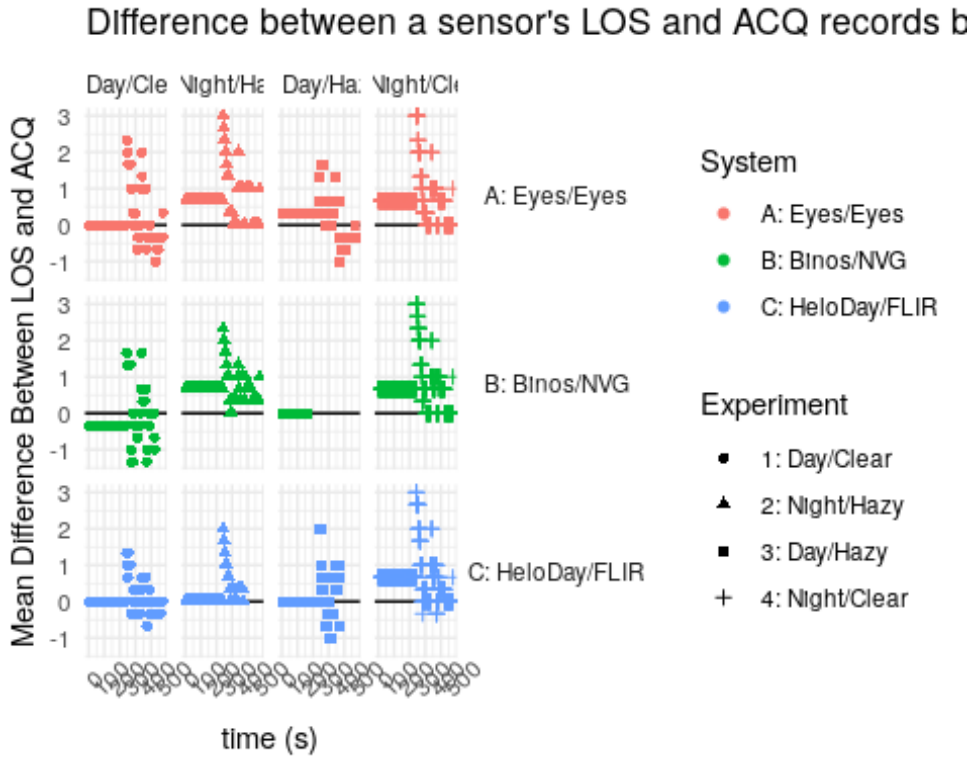
```
graphData <- modSim::Step3_multiDesingPointAndSensorDataPrep(pgConnParam = pgConnParam,
                                                             sensorForce = sensorForce,
                                                             targetForce = targetForce,
                                                             designPoint = designPoints,
                                                             sensors = sensors)
```

Calculate the Mean Difference and Graph It

This function takes the data that resulted from the previous function and calculates the mean difference between line of sights and acquisitions for sensor-target pairs by iteration. We then take that data, provide some more descriptive names for the experiments and systems, and plot it.

```
meanIterationData <- modSim::Step3_meanDiffByDesignPointAndTime(iterationData = graphData$byIt
eration)

meanIterationData %>%
  dplyr::mutate(Experiment = stringr::str_extract(string = designPoint,pattern = ".{1}$"),
                System = stringr::str_extract(string = designPoint, pattern = ".{1}(?=.{1}$)")
,
                Experiment = dplyr::case_when(
                  Experiment == "1" ~ "1: Day/Clear",
                  Experiment == "2" ~ "2: Night/Hazy",
                  Experiment == "3" ~ "3: Day/Hazy",
                  Experiment == "4" ~ "4: Night/Clear",
                ),
                System = dplyr::case_when(
                  System == "A" ~ "A: Eyes/Eyes",
                  System == "B" ~ "B: Binos/NVG",
                  System == "C" ~ "C: HeloDay/FLIR"
                )) %>%
 ggplot2::ggplot(.) +
  ggplot2::geom_hline(mapping = ggplot2::aes(yintercept = 0)) +
    ggplot2::geom_point(mapping = ggplot2::aes(x = time_s,
                                               y = diff_mean,
                                               color = System,
                                               shape = Experiment)) +
    ggplot2::theme_minimal() +
    ggplot2::labs(x = "time (s)",
                  y = "Mean Difference Between LOS and ACQ",
                  title = "Difference between a sensor's LOS and ACQ records by time step") +
  ggplot2::theme(strip.text.y = ggplot2::element_text(angle = 0),axis.text.x = ggplot2::elemen
t_text(angle = 45)) +
  ggplot2::facet_grid(System ~ Experiment)
```


Difference between a sensor's LOS and ACQ records b
Day/Cle
Night/Ha
Day/Ha.
Night/Cl
Mean Difference Between LOS and ACQ
A: Eyes/Eyes
B: Binos/NVG
C: HeloDay/FLIR
time (s)
System
A: Eyes/Eyes
B: Binos/NVG
C: HeloDay/FLIR
Experiment
1: Day/Clear
2: Night/Hazy
3: Day/Hazy
4: Night/Clear

References

[1] MSG-164, "MSaaS - Customer Perspective," in *NATO CAX Forum (online)*, Rome, 2020.

[2] Specialist Team MSG-131, "Modelling and Simulation as a Service: New Concepts and Service-Oriented Architectures," Science and Technology Organization (STO), North Atlantic Treaty Organization (NATO), Neuilly-sur-Seine Cedex, 2015.

[3] MSG-136, "Modelling and Simulation as a Service (MSaaS) - Rapid Deployment of Interoperable and Credible Simulation Environments," Science and Technology Organization (STO), North Atlantic Treaty Organization (NATO), Neuilly-sur-Seine, 2018.

[4] S. R. Goerger, A. M. Madni and O. J. Eslinger, "Engineered Resilient Systems: A DoD Perspective," in *Conference on Systems Engineering Research*, Redondo Beach, 2014.

[5] D. Sullivan, E. Colbert and J. Cowley, "Mission Resilience for Future Army Tactical Networks," in *Resilience Week (RWS)*, Denver, 2018.

[6] MSG-136, "Modelling and Simulation as a Service, Volume 1: MSaaS Technical Reference Architecture," North Atlantic Treaty Organization (NATO) Science and Technology Organization (STO), Neuilly-sur-Seine Cedex, 2019.

[7] S. M. Biemer, D. A. Flanigan and W. C. Starr, *462-M8-P1: Lecture 9A - Trade Studies,* Baltimore: Johns Hopkins University, 2015.

[8] Defense Acquisition University, "Systems Engineering Process," [Online]. Available: https://www.dau.edu/acquipedia/pages/articledetails.aspx#!204 . [Accessed 20 08 2020].

[9] I. Sommerville, Software Engineering, 10th edition, Essex: Pearson Education Limited, 2016.

[10] MSG-136, "Operational Concept Document (OCD) for the Allied Framework for M&S as a Service," 2019.

[11] National Oceanic and Atmospheric Administration (NOAA), NOAA Trade-Space Analysis Guide (TSAG), Washington D.C.: National Oceanic and Atmospheric Administration (NOAA), 2012.

[12] Y.-T. T. Lee, "Information Modeling: From Design to Implementation," *IEEE Transactions on Robotics and Automation,* 1999.

[13] M. Capuccini, A. Larsson, M. Carone, J. A. Novella, N. Sadawi, J. Gao, S. Toor and O. Spjuth, "On-deman virtual research environments using microservices," *PeerJ Comput,* 2019.

[14] J. A. Novella, P. E. Khoonsari, S. Herman, D. Whitenack, M. Capuccini, J. Burman, K. Kultima and O. Spjuth, "Container-based bioinformatics with Pachyderm," *Bioinformatics,* pp. 839-846, 2018.

[15] A. Randal, "The Ideal Versus the Real: Revisiting the History of Virtual Machines and Containers," *ACM Computing Surveys,* 2020.

[16] M. Souppaya, J. Morello and K. Scarfone, "Application Container Security Guide, NIST SP 800-190," National Institute of Standards and Technology (NIST), Washington D.C., 2017.

[17] Department of Defense Chief Information Officer, "DoD Enterprise DevSecOps Reference Design," Department of Defense, Washington, D.C., 2019.

[18] B. Patel, "MSG-168 Lecture Series on Modelling and Simulation as a Service (MSaaS): 1, 5, 7," Neuilly-sur-Seine Cedex, 2019.

[19] K. Ford, "MSG-168 Lecture Series on Modelling and Simulation as a Service (MSaaS): 2 - Overview of MSaaS Concept to Ecosystem," Neuilly-sur-Seine Cedex, 2019.

[20] C. Small, G. Parnell, E. Pohl, S. Goerger, M. Cilli and E. Specking, "Demonstrating set-based design techniques: an unmanned aerial vehicle case study," *Journal of Defense Modeling and Simulation: Applications, Methodology, Technology,* pp. 1-17, 2019.

[21] A. J. Nuss and T. D. Blackburn, "Toward Resilience as a Tradable Parameter During Conceptual Trade Studies," *IEEE Systems Journal,* pp. 3393-3403, 2018.

[22] Z. Wade, S. Goerger, G. Parnell, E. Pohl and E. Specking, "Incorporating Resiliene in an Integrated Analysis of Alternatives," *Military Operations Research,* vol. 24, no. 2, pp. 5-16, 2019.

[23] P. Mell and T. Grance, "NIST Special Publication 800-145: The NIST Definition of Cloud Computing," National Institute of Standards and Technology (NIST), Gaithersburg, 2011.

[24] L. Candela, D. Castelli and P. Pagano, "Virtual Research Environments: An Overview and a Research Agenda," *Data Science Journal,* vol. 12, pp. GRDI75-GRDI81, August 2013.

[25] T. L. J. Ferris, "A Resilience Measure to Guide System Design and Management," *IEEE Systems Journal,* pp. 3708-3715, 2019.

[26] technopedia, "Data Sandbox," 12 May 2020. [Online]. Available: https://www.techopedia.com/definition/28966/data-sandbox-big-data. [Accessed 13 November 2020].