REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
11/01/2018	Technical Report	

4. TITLE AND SUBTITLE
The Information Exchange Services Matrix and Supporting Services: Issue Paper in Support of Developing the Information Exchange Services Matrix

5a. CONTRACT NUMBER
FA8702-19-C-0001

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Andreas Tolk; David L. Prochnow; Omar Valverde

5d. PROJECT NUMBER
03194TS0MS

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102

8. PERFORMING ORGANIZATION REPORT NUMBER
PRS-18-4535

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
United States Air Force (AFLCMC)

10. SPONSOR/MONITOR'S ACRONYM(S)
AFLCMC

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The MITRE Corporation is developing an experimentation, rapid prototyping, and concept exploration capability to support testing, training, and acquisition decisions for the next generation of systems in close collaboration with SIMAF. In order to ensure best support of the prototypical development and its transition in later phases, the conceptual and design principals are exchanged with industry partners and academic support organizations. This issue paper captures these principles and addresses the underlying theory, the resulting architectural recommendations for JSE, the description of services belonging to the Information Exchange Service Matrix (IESM), and the concept of operations.

15. SUBJECT TERMS
Modeling and Simulation; Business Process Engineering; Education and Training (General); rapid prototyping; Joint Simulation Environment; acquisition; training; United States Air Force; JSE; testing; experimentation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE		53	Susan Carpenito
					19b. TELEPHONE NUMBER *(Include area code)* 781-271-7646

MITRE Product: MP180872

Andreas Tolk
David L. Prochnow
Omar Valverde

Hampton, VA
November 2018

Sponsor: Air Force Simulation and Analysis Facility
Contract No.:
FA8702-18-C-0001

This technical data was produced for the U. S. Government under Contract No. FA8702-18-C-0001, and is subject to the Rights in Technical Data-Noncommercial Items Clause DFARS 252.227-7013 (FEB 2014)

The Information Exchange Services Matrix and Supporting Services

Issue Paper in Support of Developing the Information Exchange Services Matrix

This page intentionally left blank.

Executive Summary

The United States Air Force (AF) is constructing its next generation enterprise training and test capability to serve as the simulation infrastructure for the next generation of their systems, the Joint Simulation Environment (JSE). The AF Life Cycle Management Center Simulation and Analysis Facility (SIMAF) team, with significant help from other partners and developers, plays a leading role in developing the architecture for the required capabilities.

The MITRE Corporation is developing an experimentation, rapid prototyping, and concept exploration capability to support testing, training, and acquisition decisions for the next generation of systems in close collaboration with SIMAF. In order to ensure best support of the prototypical development and its transition in later phases, the conceptual and design principals are exchanged with industry partners and academic support organizations. This issue paper captures these principles and addresses the underlying theory, the resulting architectural recommendations for JSE, the description of services belonging to the Information Exchange Service Matrix (IESM), and the concept of operations.

This page intentionally left blank.

Table of Contents

List of Figures

This page intentionally left blank.

1 Introduction

The United States Air Force (AF) is constructing its next generation enterprise training and test capability to serve as the simulation infrastructure for the next generation of their systems, the Joint Simulation Environment (JSE). The AF Life Cycle Management Center Simulation and Analysis Facility (SIMAF) team, with significant help from other partners and developers, plays a leading role in developing the architecture for the required capabilities.

In partnership with SIMAF, The MITRE Corporation is developing an experimentation, rapid prototyping, and concept exploration capability to support testing, training, and acquisition decisions for the next generation of systems. These MITRE tasks include the development of an Information Exchange Services Matrix (IESM). The IESM forms the core of a new simulation integration framework and information broker and is a critical enabling component of the JSE. Using general simulation interoperability concepts based on state-of-the-art integration technology, it will meet AF objectives by:

- Interfacing live systems with manned and constructive simulations of aircraft, sensors, weapons, threats, command and control nodes, communication and data networks, and dynamic weather conditions at fidelity levels and data rates sufficient for realistic testing, training, and analysis.
- Providing key simulation services for time management, communications, and data transformation across multiple scenario domains (air, space, surface) in a variety of atmospheric conditions (day, night, clear and adverse weather).
- Allowing simulation configuration and orchestration without the need to involve a simulation specialist.
- Facilitating data collection and analysis.
- Enabling multi-level security (MLS) solutions to support localized and distributed simulation operations within the JSE.

In addition to creating technical specifications and reference implementations, MITRE is conducting in-depth research on specific topics to ensure the solutions proposed reflect the state of the art. The technical documents that MITRE will produce will focus on selected solutions, rather than on the full range of possible alternatives, and therefore will not cover all the research findings. To preserve this research and provide additional information justifying the selections, MITRE is capturing its findings in the form of “issue papers” that provide deeper insight into specific topics.

This issue paper summarizes MITRE’s research on the use of services to provide a distributed, composable simulation environment as envisioned for the JSE. It focuses on the core services of the IESM that support information exchange during the execution of a simulation event; additional exercise services that support preparing, conducting, and evaluating the event; and common services that provide for the consistent representation and computation of critical entities and processes within the simulation event. The paper defines the categories and services and describes their interrelations.

2 Underlying Theory

The envisioned next generation enterprise training and test capability shall extend and improve already available technical solutions and capabilities. It shall not provide merely a point solution for a single event but shall provide an architecture that serves as the framework for the reuse of existing solutions and allows the seamless integration of new solutions, based on the characteristics of scalability, flexibility, adaptability, modularity, and configurability, as described in Figure 1.

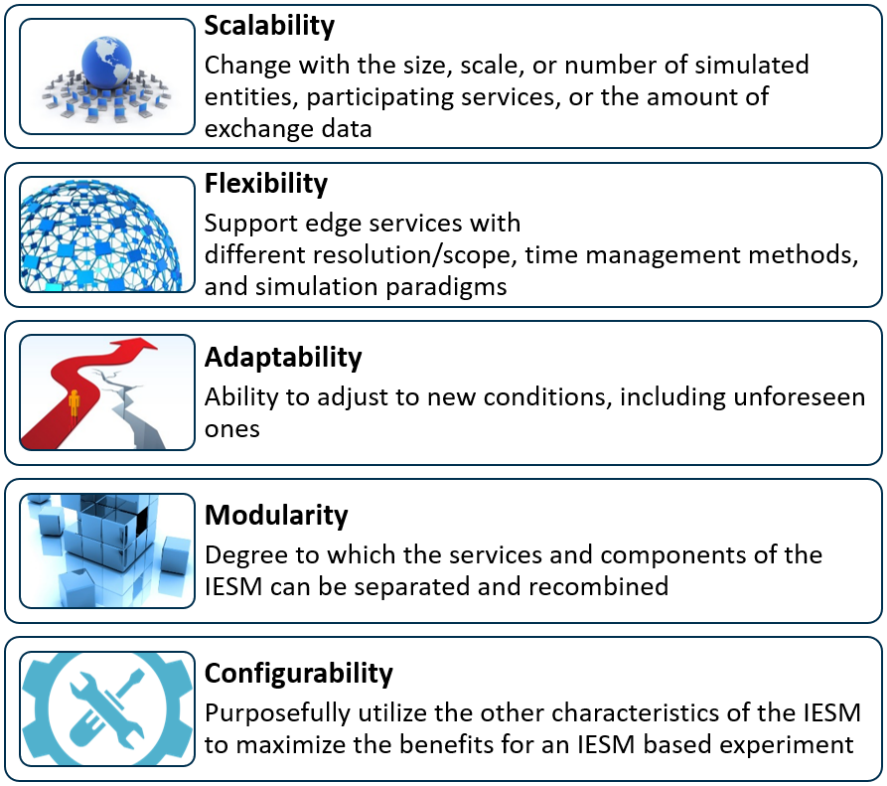

Figure 1: Architecture Characteristics of the JSE and the IESM

Software services are reusable components that provide functionality as needed to accomplish various tasks. The use of composable simulation services and information exchange services to construct flexible compositions that deliver the necessary capabilities for operational testing offers a technically mature foundation for future solutions. The core of the recommended solution is the IESM, which comprises a set of core services supported by a set of exercise services and common services.

2.1 Service-based Simulation Experiments

The US Modeling and Simulation Coordination Office (MSCO) succeeded the US Defense Modeling and Simulation Office (DMSO) as the administrative organization that coordinates modeling and simulation (M&S)-related activities within the US Department of Defense (DoD). DMSO supported the development of the Verification, Validation and Accreditation (VV&A) Recommended Practice Guide (RPG) that was first published in 2006

and is still frequently updated and used.[1] As the document is freely available, many other organizations employ the recommended practices as well. Of interest in this issue paper is the general problem-solving process, shown in Figure 2. This general process defines the specific processes related to the application of M&S-based solutions that contribute to solving user problems.

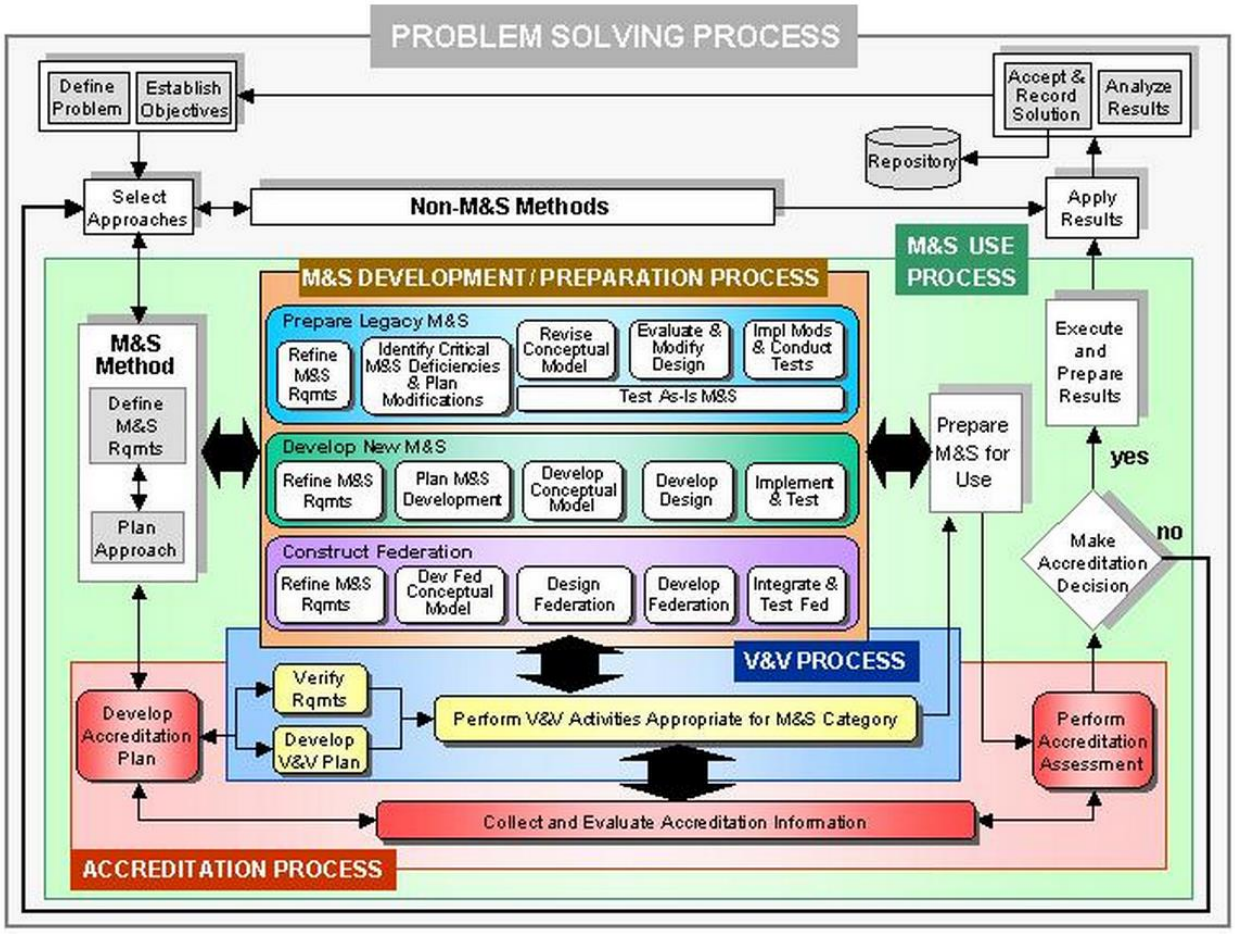

Figure 2: General Problem-Solving Process within the US DoD

From the processes and accompanying best practices, DoD organizations derived the following tasks that simulation engineers must perform when providing capabilities via services to support a simulation experiment. As recommended in many best practices, such as the *NATO Code of Best Practice for Command and Control Assessment*[2] or the *Guide for Understanding and Implementing Defense Experimentation*,[3] all such efforts must be rooted in the operational problem of the sponsor. Therefore, starting with the operational mission and deriving technical solutions from these needs is good practice for test and training as well.

[1] Accessible at https://vva.msco.mil/.
[2] Alberts, Dean S., et al. 2002. *NATO Code of Best Practice for Command and Control Assessment*. Washington DC: CCRP Press.
[3] Bowley, Dean, et al. 2006. *Guide for Understanding and Implementing Defense Experimentation (GUIDEx)*-Version 1.1. The Technical Cooperation Program (TTCP).

1. **Define the scope of the experiment:** In this step, simulation engineers identify the mission, the required systems, and their activities and rules of engagement. This includes defining the friendly and opposing forces, including their activities and rules as well as the environment in which the experiment will take place. These activities enable creation of an operational blueprint of the experiment, independent of the simulation environment.
2. **Identify applicable services:** The engineers then map the blueprint of the operational mission to a conceptual model of how simulation systems can be used to support the mission. They develop this model to define key variable relationships, including what will be varied (independent variables), what will be measured (dependent variables), and what will be held constant (control variables). Using existing documentation or otherwise accessible knowledge about simulation services available to support the experiment, the engineers identify all services that support at least a significant subset of the mission. These services become candidate elements of the services that will provide the capabilities for the simulation experiment.
3. **Select the collection of the best services:** Using the candidate services, engineers select the optimal composition to provide full mission coverage. They can pursue multiple objectives when making their choices, such as minimizing the interfaces that must be created or maintained in the composition, maximizing the fidelity of the participating simulation services, minimizing costs, accommodating runtime constraints, ensuring accessibility of solutions, etc. Steps 2 and 3 lead to the simulation blueprint of the experiment that shows which simulation services will simulate which part of the operational mission, which variables they must exchange, etc.
4. **Compose the selected services:** In this technical integration phase, engineers determine how to interface the services with the supporting infrastructure (provided by the IESM for the JSE), how different data representations and formats will be mapped and mediated, how information that must be provided to other services will be made accessible, and how information received from other services will be accepted and mapped to concepts represented within the receiving services.
5. **Execute the experiment:** Once the services are composed, engineers can conduct the simulation experiment. Whether or not the experiment is performed in real time, the execution of the various services must be synchronized and managed. The simulation infrastructure must ensure that the services receive all the information they require to perform their functions, and receive it on time. Moreover, services should receive only the required information to avoid wasting resources, such as bandwidth and computing power, on unnecessary data exchanges. Further, the services must ensure security of any sensitive or classified information.
6. **Collect data needed to evaluate experiment:** Collecting data is pivotal for runtime or post-event analysis. Performed simultaneously with the experiment, the data collection activity ensures that all required data are available and accessible. This requires simulation engineers to construct a plan for collecting data exchanged

during distributed experiments – easily accessible via the infrastructure – as well as access to required data that was not exchanged. All data must be tagged with metadata to enable an unambiguous evaluation process.

7. **Conduct the evaluation:** This last step enables the engineers to evaluate how well the conducted experiment meet sponsor needs. A proper evaluation includes determining how many repetitions are needed to address stability – distribution within a solution point – and sensitivity of solutions – evaluating the close neighborhood.[4] During the evaluation process, engineers must also ensure the security of any classified information that gets accessed by evaluating experts, e.g., by ensuring that the need-to-know principle is applied. It is good practice to collect the results and populate a result repository with information on how data were collected, which services were used, etc.

These processes are congruent with the steps recommended by the IEEE 1760 Standard on the Distributed Simulation Engineering and Execution Process (DSEEP).[5] In addition, the academic community dealing with the philosophy of science identified two potential challenges that organizations must consider in determining the validity of any simulation experiment and its evolution.

- The *epistemological* challenge centers on the limits of an experiment that stem from the conceptual model: does the model capture all relevant concepts, properties, relations, and processes, and capture them at the correct abstraction level? Because the conceptual model describes the "reality" to be captured in the simulation, only concepts included in the model can be evaluated. If important aspects are overlooked, the simulation systems cannot take them into consideration.
- The *hermeneutical* challenge centers on the interpretation of the results themselves. When interpreting the results of the simulation experiment, engineers must be guided by the conceptual model underlying the experiment. It is human nature to interpret results in the light of one's favorite world view, often resulting in unconscious bias. Interpretations should only use assumptions and constraints that are captured within the conceptual model, because all other explanations extend beyond the simulation experiment and consider concepts outside of the model's scope.

The following sections establish the importance of the conceptual model beyond these theoretic insights and show the practical implications for the JSE.

2.2 Conceptual Alignment

Simulation engineers must address conceptual challenges as well as the technical challenges described in Section 2.1. These conceptual challenges arise from the observations that models are purposeful, task-driven abstractions and simplifications of a perception of reality. They are developed (purposefully) to fulfill a given objective, such as

[4] Additional information is described in Tolk, Andreas, "Stability and Sensitivity Measures for Solutions in Complex, Intelligent, Adaptive and Autonomous Systems," Symposium for Modeling and Simulation of Complexity in Intelligent, Adaptive and Autonomous Systems; Spring Simulation Multi-Conference 2016, Palo Alto, CA.
[5] IEEE 1730-2010 DSEEP. IEEE Standards. IEEE Piscataway, New Jersey.

testing a system or training users under a certain set of conditions (task-driven). Using the abstraction level appropriate to the task (abstraction) and excluding all unimportant concepts (simplification) results in a properly formed conceptual model. However, this model is inevitably based on the knowledge and world view of the implementor (perception of reality), which can be shaped by physical, legal, and cognitive constraints.[6] The same real-world referent can lead to a multitude of conceptual representations.

When implementing these models as computer simulations, engineers also add diversity due to choices of modeling types, programming languages, and numerical heuristics used within the implementation.[7] As a result, even if the same model is used, the implementations can differ significantly, and the same real-world referent can be represented by different models, depending on which tasks the models had to perform.

Making sure that these differences in conceptualization and implementation do not lead to conflicts and inconsistencies in a composition of such independently developed solutions defines the task of conceptual alignment. No *technical solution can overcome a conceptual conflict.* To address these challenges, Page et al.[8] introduced three categories of attributes needed to ensure correct interconnection and composition of simulation-based solutions. In recent years, they have refined these categories to reflect the latest research results.

- *Integratability* applies to the physical/technical realms of connections between systems, which include hardware and firmware, protocols, networks, etc. Integratability ensures that signals can be exchanged and correctly mapped to symbols used for the computational manipulation within the components.
- *Interoperability* applies to the software and implementation details of operations between systems, which include the exchange of data elements via interfaces, the use of middleware, mapping to common information exchange models, etc. Interoperability ensures that data can be exchanged between components and used in the receiving system.
- *Composability* applies to the alignment of issues at the modeling level. Composability ensures that truth is consistently represented in all participating components.

In the electrical engineering (EE) domain, the role of integratability and interoperability are sometimes inversed, as the objective of EE is to provide solutions that are ready to be integrated as building blocks into a larger electrical system. In contrast, the definitions by Page et al. focus on the simulation engineering domain, where such integratable EE solutions build the basis of the infrastructure. The interoperability challenges addressed in the above definition emerge from the simulated domain, not the EE domain.

---

[6] Tolk, Andreas. 2013. "Interoperability, Composability, and Their Implications for Distributed Simulation - Towards Mathematical Foundations of Simulation Interoperability." IEEE/ACM/SCS Distributed Simulation – Real Time (DS-RT) Conference, Delft, The Netherlands.

[7] Oberkampf, William L., Sharon M. DeLand, Brian M. Rutherford, Kathleen V. Diegert, and Kenneth F. Alvin. 2002, "Error and Uncertainty in Modeling and Simulation." *Reliability Engineering & System Safety* 75 (3): 333–357.

[8] Page, Ernest H., Richard Briggs, and John A. Tufarolo. "Toward a Family of Maturity Models for the Simulation Interconnection Problem." *Proceedings of the Spring Simulation Interoperability Workshop*. Vol. 1, pp. 1059-1069, IEEE Computer Society, 2004.

Tolk and Muiguira[9] created the Levels of Conceptual Interoperability Model (LCIM) to structure discussions on the integration of infrastructure, interoperation of simulations, and composition of models. Since its introduction in 2003 this framework has been extended and applied multiple times in many application domains, from M&S applications to the Electric Smart Grid, the Internet of Things, and more. Figure 3 shows the model levels.

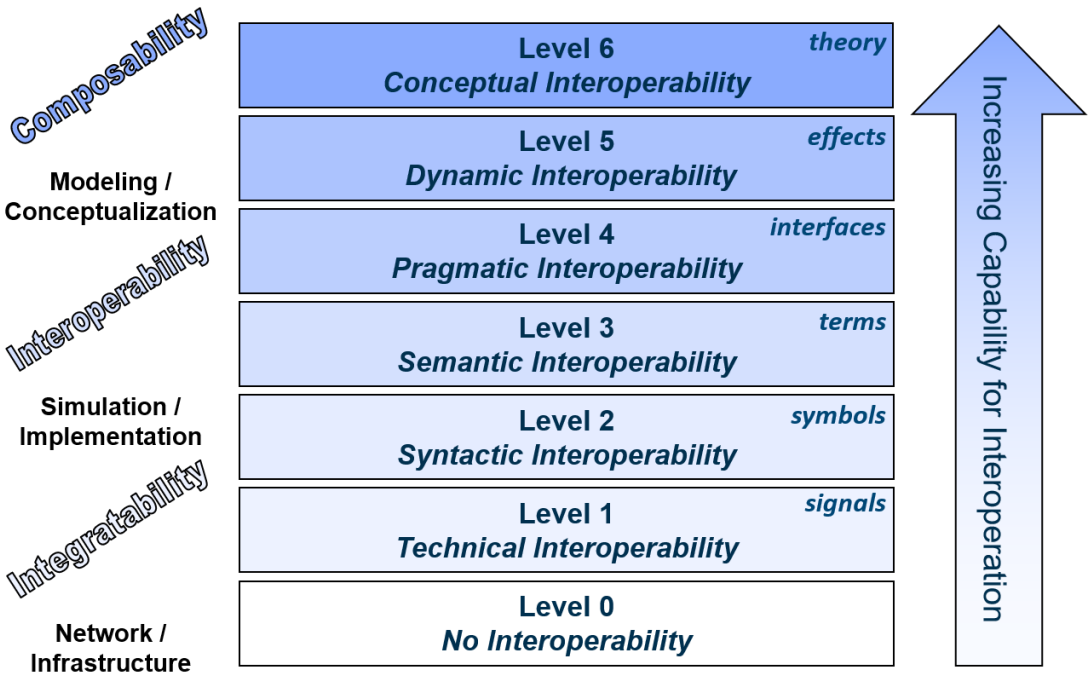

Figure 3: Levels of Conceptual Interoperability Model (LCIM)

The layers shown in the figure, which are used to cluster problem classes of integratability, interoperability, and composability, can be described as follows.

- Level 0: Stand-alone systems are not connected and have no interoperability.
- Level 1: The technical layer deals with infrastructure and network challenges, and forms the foundation of any ability to exchange information. This level ensures systems can exchange signals, the carriers of information.
- Level 2: The syntactic layer deals with protocols. It ensures the use of common symbols within the receiving components.
- Level 3: The semantic layer fosters common understanding through use of common terms. These terms can be captured in a common namespace and even in a taxonomy.
- Level 4: The pragmatic layer recognizes the patterns in which data are organized for information exchange, such as the inputs and outputs of procedures and methods

[9] Tolk, Andreas, and James A. Muguira. "The Levels of Conceptual Interoperability Model." *Proceedings of the Fall Simulation Interoperability Workshop*. IEEE Computer Society, 2003.

provided by the components. This results in well-understood interfaces used to invoke services by providing the necessary information in the right structure.

- Level 5: The dynamic layer provides transparency of the receiving components, including timing challenges and concepts needed to harmonize the processes. It ensures clear understanding of effects resulting from calling a component.
- Level 6: Finally, the conceptual layer handles assumptions, constraints, and simplifications. This layer ensures that the participating components represent facts either equally or not at all, thus ensuring a common "theory" of the mission.

The LCIM has been applied to identify gaps (descriptive use) as well as to propose common solutions (prescriptive use) for simulation compositions. The mathematical branch of Model Theory provides the academic foundation.[10]

The elusiveness of the conceptual interoperability level often presents a challenge for practitioners who must find ways to interconnect model-based systems using common terms to describe concepts such as the probability of striking a target, or the amount of energy reflected. However, terms are often homonyms, because they were derived under different assumptions that applied in a specific context. An example is the likelihood that an air attack will succeed in striking a ground target. This likelihood is influenced by many factors, including weather, terrain where the target is located, degree of fortification and camouflage used, protective measures such as air defense, and others. If, for example, the model already considers the presence of air defense when computing the probability of a hit, and then explicitly models the effects of air defense in addition to the target systems, the effect of air defense is counted twice and the effect is over-represented, leading to incorrect assessments.

The effect of battle management connectivity to enable next-generation sensor fusion and advanced network-enabled capabilities is pivotal for the performance of the targeted systems, as these systems significantly increase their air-to-air and air-to-ground combat capabilities. Simulation environments developed in support of earlier generation systems may not represent all the concepts necessary to train and test next generation of currently developed and future systems.

2.3 Implications for the JSE/IESM Efforts

To support the processes identified so far, models must capture the definitions of relevant terms in computationally accessible form, e.g., by using ontological descriptions that produce machine-readable artifacts. Formal models, captured using standards such as UML, SysML, OWL, or sufficiently powerful alternatives, must ensure the necessary transparency required for composability.

- Services must expose their entities, properties, relations, and processes, including the time constraints.

[10] Diallo, S.Y., Padilla, J.J., Gore, R., Herencia-Zapana, H. and Tolk, A., 2014. "Toward a Formalism of modeling and Simulation Using Model Theory." *Complexity*, 19(3), pp. 56-63.

- The supported mission that constitutes the operational foundation for the test or training event must also be captured as a formal model that describes who is doing what, when, and where.
- Constraints on computational resources, including time, access points, and routes, must be captured accordingly to allow machine-supported configuration and scheduling of services.

AF and MITRE engineers should use model-based systems engineering and automatic model transformation between alternative representations of conceptually equivalent implementations in creating the JSE and IESM. Success depends not only on applying these principles in new developments, but also on documenting every contributing legacy component with the same rigor.

3 Architectural Recommendations

The underlying vision for the current research and development efforts addresses the requirement to support higher generations of weapon systems[11] with a scalable, flexible, adaptable, modular, and configurable simulation infrastructure. MITRE identified the use of composable services as the current best practice to fulfill these requirements.

- Depending on the size, scale, or number of simulated entities and participating services, or the amount of data to be exchanged, the number of services providing the needed functionality can vary. Composable services allow engineers to add or remove services as needed.
- Different service instantiations can provide similar functionality to services that have different resolution/scope, time management needs, and supported simulation paradigms.
- Should new or unforeseen conditions occur, only those services affected by those conditions have to be updated, allowing rapid and efficient adaptation of the solution to the new environment, tasks, missions, threats, etc.
- Services are modular by design, allowing engineers to separate and recombine them based on the principles of loose coupling in the technical domain and composability in the conceptual domain.
- All properties captured so far lead to a fully configurable solution that provides all needed functionality in a most efficient way.

Many currently available solutions do not allow engineers to instantiate only the services needed. Instead, all services that come with the chosen simulation interoperability infrastructure must be installed, whether or not they are used in an experiment. Similarly, many solutions are not scalable in terms of the number of supported entities.

3.1 Joint Simulation Environment as a Composition of Services

The JSE shall provide an Enterprise Training and Test capability. This results in requirements that affect several layers, leading to the composition of various service categories as depicted in Figure 4.

JSE is not an isolated effort. The AF has invested significant resources in the development of various simulation solutions, including flight simulators for training and testing, constructive simulation systems such as the Air Warfare Simulation (AWSIM) model and the Extensible Architecture for Analysis and Generation of Linked Simulations (EAAGLES), highly detailed technical simulations, and others. All these simulation solutions provide valuable functionality that may contribute to fulfilling requirements for a particular simulation experiment. Therefore, these existing legacy solutions should be accessible so that engineers can draw upon them to provide their functionality for a new simulation

[11] Hill, Raymond R., Andreas Tolk, Douglas D. Hodson, and Jeremy R. Millar: "Open Challenges in Building Combat Simulation Systems to Support Test, Analysis and Training." *Proceedings of the 2018 Winter Simulation Conference*, Piscataway, NJ: IEEE

experiment. These reusable legacy simulation solutions fall into the category of edge services that are connected to the IESM.

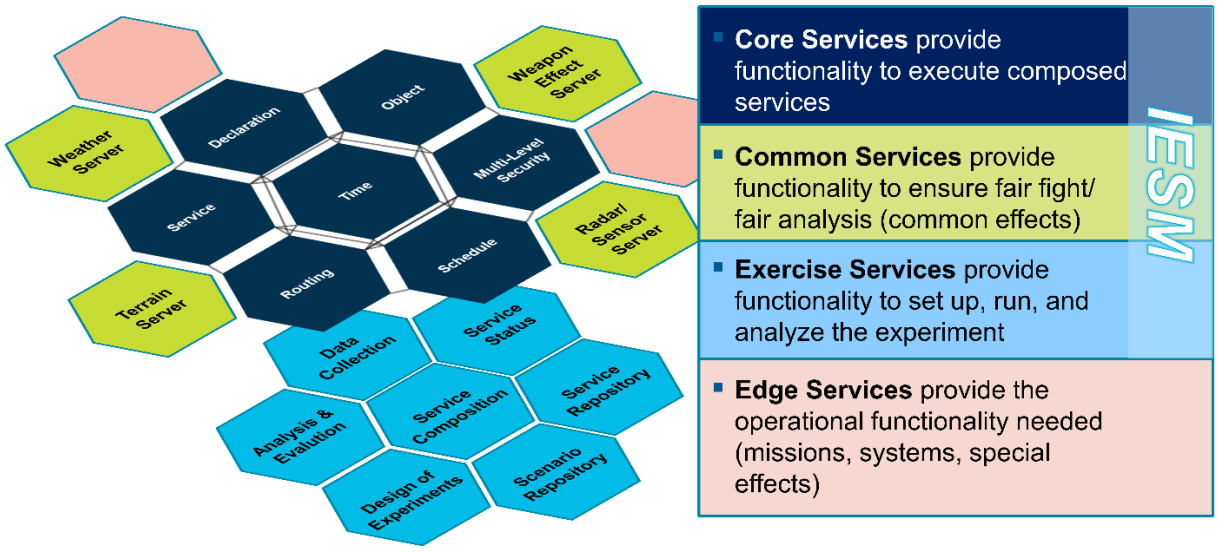

Figure 4: IESM and Supporting Services

Another important set of edge services consists of rehosted Operational Flight Program (OFP) code collections that enable greater fidelity, in particular for blue systems. These collections can be full OFP for the systems or components to be tested, such as new sensors, etc. These services also provide functionality to the simulation experiment; they represent platforms or components with the highest fidelity possible by reusing the original code, and sometimes even hardware, used in the operational environment.

The IESM connects these two categories of edge services – legacy simulation solutions and OFPs and related components. As noted, edge services are not part of the IESM, but they do belong to the JSE. As a rule, they provide the operational functionality required by the training and test communities, while the IESM provides the technical functionality that simulation engineers need in order to use and reuse edge solutions in a composition of services that builds the JSE.

3.2 IESM and Supporting Services

Legacy simulation solutions and OFPs were not necessarily developed with the JSE requirements in mind. They often incorporate documented application programming interfaces (APIs) that allow the technical coupling of these solutions via a common infrastructure. As noted in section 2.2, more transparency of components is needed. Interface specification are not sufficient.

3.2.1 IESM Service Categories

The IESM was designed to address all challenges that simulation engineers encounter when composing JSE services to conduct a simulation experiment for training or testing. To meet its goals, the IESM comprises three categories of services, as depicted in Figure 4.

- *Core services*, also known as primary services, provide the functionality needed to conduct a simulation experiment, and are necessary for the planning, development, management, execution, and evaluation of distributed simulation events. They supply the proper infrastructure that ensures secure and timely communication among the other services to conduct simulation-based experiments and other events.
- *Common services* perform functions that are pivotal for the simulation of the supported domain, ensuring that all participating components have a consistent conceptualization and implementation of common concepts such as terrain, weather, weapon effects, cyber effects, communications, space, and more. They ensure a "fair fight" within the simulation composition.
- *Exercise services* support the planning, development, management, execution, and evaluation of distributed simulation events by delivering secondary functionality, such as collecting data and creating repositories and other enablers. Exercise services provide the functionality needed to set up a simulation experiment, collect data during the execution, generate useful meta-information on the status of the network and all components, and evaluate the experiment following execution.

The subsections below describe all three categories in more detail.

3.2.2 IESM Core Services

Core services are intended primarily to ensure that *all required information* is exchanged between the other services, *only the required information* is exchanged, and the exchanges occur *securely* and in a *timely* manner. To accomplish this, MITRE recommends use of core services that fall into seven categories. MITRE derived these categories from an evaluation of current simulation interoperability protocols and their implementations, with a focus on the Distributed Interactive Simulation (DIS) protocol,[12] the High Level Architecture (HLA) Runtime Infrastructure (RTI),[13] and the Test and Training Enabling Architecture (TENA).[14]

- *Service management*: These services provide information about which simulation services are accessible and online (i.e., currently participate in an ongoing exercise). If necessary, the IESM can launch more than one service manager to provide other IESM services with status information on services that could potentially participate, scaling to the number of participating edge services.
- *Declaration management*: These services provide information about which components will produce and which components will consume information objects

[12] IEEE 1278 Distributed Interactive Simulation (DIS), published in several volumes.
[13] IEEE 1516 Standard for Modeling and Simulation (M&S) High level Architecture (HLA), published in several volumes, and accomplished by IEEE 1730 Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP), published in two volumes.
[14] TENA has not been standardized but provides many good solution idea: http://www.tena-sda.org.

to be exchanged, and which operationally relevant services the IESM provides to the simulated operation. They work closely with the service managers and allow them to initialize publish-subscribe, push-pull, and alternative information exchange methods based on availability and interest.

- *Object management*: In contrast to declaration management services, these services supply information on simulated objects that are currently shared, including which edge service initiated the object and which service is currently in charge of such distributed information objects. These services also provide transient storage for information objects that are used frequently and should be provided immediately to a requesting service if the operation is time critical. They also perform data mediation services. Several object managers may be needed to scale the IESM to the desired number of simulated objects.
- *Time management*: These services synchronize the execution of all participating simulation services, including services that operate faster and slower than real time. Details of time management services and possible solutions are captured in another issue paper.[15]
- *Routing management*: These services enable transfer of tailored data between participating services via configurable communication channels, including, but not limited to, the specifications defined by the Interface Control Documents (ICDs) for the rehosted OFPs. The services also ensure that only required information is submitted, and only information that is not already available in the transient storage is requested from the producing simulation service. Routing managers can scale and adapt to the number of required routes and constraints for such routes. The concept of multiple parallel but synchronized routes is new to the simulation community and has not been addressed in any of the current simulation interoperability standards.
- *Schedule management*: When operating in high-performance environments, it may become necessary to adapt the order in which simulation and IESM services are executed and to produce required information objects to ensure real-time availability. Schedule managers address this challenge. In addition to knowing the technical state of all participation simulation services, these services may also require mission awareness. This topic is also new to the simulation interoperability community.
- *Multilevel security management*: When simulation services and audiences with different security classification/clearance levels participate in the same simulation experiments, the accessibility and visibility of data and services depend on the user role and clearance/classification level of the user and/or the service. By obfuscating and filtering/masking information objects and service performance accordingly these services provide cyber security against illegal access or modification of the configurations and data. Details of MLS management services and possible solutions are captured in another issue paper.[16]

[15] MITRE Product (MP180044) Time Management Services, January 2018.
[16] Tolk, Andreas, and David Prochnow, "Multilevel Security Services," MITRE Product (MP180272), June 2018.

The appendix to this issue paper describes the many detailed interrelations among these service categories. All services can be instantiated as often as needed to scale to the requirements of the simulation experiment. Furthermore, they can be adapted to support new requirements, different levels of resolution, etc. If they are not needed, the IESM does not instantiate them, as they all contribute their functionality only when needed by the IESM.

3.2.3 IESM Exercise Services

In contrast to the core services, the list of exercise services shown in Figure 4 represents only a subset. As noted, they deliver secondary functionality needed for the planning, development, management, execution, and evaluation of distributed simulation events. Exercise services should provide a framework that allows the integration of all services needed to support the simulation experiment, including the use of operational data where desirable and possible within classification constraints.

The military training and test communities already apply comparable solutions providing such functionality, but often not in the scalable, flexible, adaptable, modular, and configurable way envisioned for the JSE. Examples are the Joint Operation Planning and Execution System (JOPES)[17] and the Joint Training Data Services (JTDS).[18]

The IESM can apply many tools for network configuration, management, and monitoring, for launching services that enable remote start or termination of distributed computing systems, and more. It can also use already developed or openly available three-dimensional (3D) mapping and virtual globe platforms to display the simulated situation. Of particular interest is the possibility of using operational planning tools to plan and initialize simulation experiments. The planner responsible for a training or testing event should not have to be concerned with the simulation infrastructure but should be able to design the mission underlying the training or test with the tools he or she is used to working with.

Data collection can pose an additional challenge, depending on the metrics against which the decision maker wishes to evaluate the success of the simulated mission. If IESM services exchange the data required for the metrics, accessing the data is easy and only requires coordination between the exercise services and the core services. However, if the data are not exchanged, the exercise service may become a new requester. Meeting this request is not especially challenging if the data can be provided by the interfaces of the service that owns the data, but if the data are hidden deep inside the implementation details of the owning service, the data collection services must either select alternative data, or some significant changes in the simulation service will be needed.

3.2.4 IESM Common Services

While core services and exercise services add functionality needed to enable the composition of edge services to set up, conduct, and evaluate a simulation experiment, the common services replace functionality already provided by individual edge services with a common conceptualization and implementation of that functionality. Therefore, the use of

[17] Assessible at https://publicintelligence.net/joint-operation-planning-and-execution-system-vol-1/3639/.
[18] Assessible at https://jtds.jten.mil/jtds/.

common services will likely require code changes in the edge services. If, for example, the IESM uses a common weapon effect service, it must disable the internal computation of effects and instead implement a call on the externally provided function. Also, all internal computations that use the effects of an engagement as a parameter must be revisited to ensure that they are performed in the right order and their validity constraints are not violated, and to execute other related coordination tasks.

The use of common services requires a high level of collaboration among the providers of the IESM, the edge services, and the simulation sponsors and stakeholders. The reason is that common services represent the exception to the rule that the edge services provide the operational functionality required by the training and test communities.

3.3 Composer, Orchestrator, Conductor, and Performer

The users of these service compositions, predominantly simulation engineers, should employ a set of tools that support simulation planning, selection of services, conceptual alignment, and other important steps. The tools should enable engineers to hide technical details while ensuring that simulation processes execute with the necessary rigor. These tools fall into four categories whose names are borrowed from the music domain: composer, orchestrator, conductor, and performer. Figure 5 shows the concept of operations.

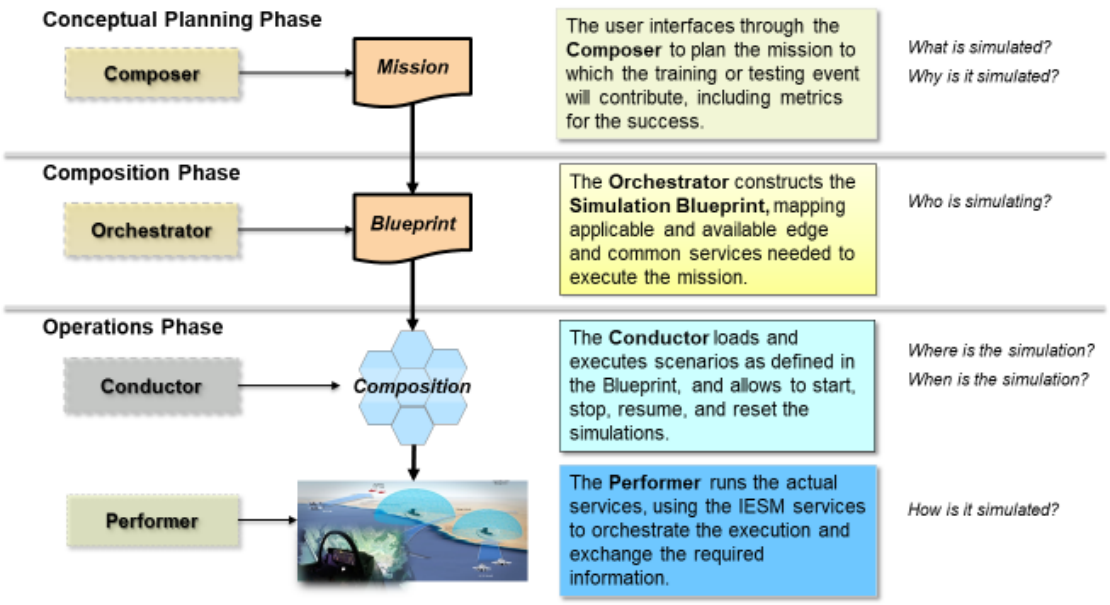

Figure 5: IESM Concept of Operations

The mapping of the steps of a simulation-based experiment to these tool groups becomes obvious when comparing the outcomes of the steps:

- Using the composer allows engineers to define the simulation-based experiment from the warfighter's perspective. It captures the mission with its constituent

systems, actions, and rules, including the opponent and the environment in which the mission is executed.

- The use of the orchestrator helps identifying simulation services that can be applied to support the mission – or part of it – and helps engineers to select the best services to support the objectives of the sponsor. The result is the blueprint for the simulation-based experiment.
- Using the conductor, users first compose the services and configure the core services needed for this composition, scale the services to provide the best performance, calibrate and instantiate the exercise services, etc. When this step is completed, the experiment is set up and can be conducted. During the experiment, the conductor directs how the performers support the execution of the experiment.
- Finally, using the performers the users execute the experiment, collect the necessary data, and enable evaluation of the results

Many simulation environments blur the line between the operational tasks to be supported (composer) and the functionality provided by the simulation tools that provide this support (orchestrator). However, engineers should clearly understand the sponsor's problem first, in which allows them to devise a solution strategy that then is executed using available methods and tools. The *NATO Code of Best Practice for Command and Control Assessment* (mentioned in section 2.1 of this issue paper) clearly states that starting a study by accepting the constraints of available tools or by preselecting the method to be applied are bad practices that often lead to suboptimal studies and recommendations. If current tools do not support all aspects of a mission, and this mission is critical for the sponsor, simulation engineers should review the gaps between what the mission requires and what the available services can provide. This allows engineers to identify capability gaps that the sponsor should address in the procurement and development process.

4 Summary and Recommendations

The architecture for JSE and IESM, which uses composable services to support training and testing, fulfills the AF requirement for a scalable, flexible, adaptable, modular, and configurable simulation infrastructure. It requires that the solution provided exhibit a high degree of transparency to ensure conceptual alignment among simulation services.

The mission planned by the military experts to conduct training and testing serves as the reference point, or operational blueprint for all decisions. This blueprint covers all important entities, properties, relations, and processes that make up the mission. It provides the operational requirements that drive the selection and configuration of the simulation services. Once simulation engineers have chosen the edge services, they can use the blueprint to decide on the services that the IESM should provide.

The selected edge services drive the way engineers scale and adapt the IESM services and determine how engineers configure all services. The same degree of transparency for the IESM services allows the reuse of already developed and available solutions that fulfill the technical requirements (e.g., existing DIS infrastructures, open source implementations of the HLA RTI, etc.).

The availability of metadata supporting these steps is essential to success. Compositions based on interface descriptions often result in conceptual inconsistencies that may make training and testing invalid. Using the same standard throughout the definition, set ups, and execution of an experiment allows the reuse of solutions and supports the use of repositories, thus facilitating future training and testing events.

A long-term goal is to bridge the dichotomy between highly efficient direct communication and configurable, standardized interfaces by making increased use of model-transformation approaches. Currently, the most efficient way to connect two services is to program an individually optimized direct route for exchanging information in the best binary form possible. Unfortunately, such individual, specialized data links can neither be reused nor generalized. Standardized, reusable approaches, on the other hand, can result in cumbersome and computationally intensive solutions due to the need for many standard-specific transformations, the use of a common information exchange format that requires additional resources, etc. The ultimate solution will employ approaches that do not standardize the information exchange itself but instead use standards on a meta-level, such as defined by the Object Management Group (OMG), and derive optimized solutions via model-transformation at the implementation level; the methods to enable such approaches remain the subjects of ongoing research.

The IESM and supporting MBSE methods will allow the US Air Force to provide the necessary infrastructure to conduct test and training for the latest generation of systems while at the same time maximizing the reuse of existing solutions and providing integration for current system testing and training concepts.

This page intentionally left blank.

Appendix A: Interrelations among the IESM Service Categories

This appendix summarizes some detailed interrelations between the various categories of IESM core services. The logical specifications presented here are neither exclusive nor complete and do not replace more detailed development specifications. Instead, they are provided to broaden understanding of how the IESM core services will collaborate to fulfill the infrastructure requirements for higher generation system simulation support.

Definitions

The list below contains informal definitions for terms used in the requirements discussions that follow:

- **Object** or **Object Instance** – A specific object with persistence. It can be created, updated, and deleted. Each object is composed of one or more data fields.
- **Object Class** – A category of objects. The class defines the allowable data fields for this type of object.
- **Event** – A one-time occurrence. Each event is composed of one or more data fields. Events do not have persistence. The High-Level Architecture (HLA) term for such an event is *interaction*.
- **Event Class** – A category of events. The class defines the allowable data fields for this type of event.
- **Object Model –** A specification of the object classes and event classes that are to be used for a simulation execution.

A-1: Logical Specification of Declaration Management Services (DMS) and Service Management Services (SMS) Relationships

Overview of DMS/SMS Relationship

Service Management Services (SMS) determine which simulation services and other edge services are accessible to the IESM and to the simulation components. This is important for several reasons:

- The IESM software can operate more efficiently by limiting the services to those essential for executing a given simulation.
- Simulation components can adapt their behavior based on the simulation services available.

System components use Declaration Management Services (DMS) to announce their intent to publish or subscribe to specific object classes and event classes, or to participate in time management as a time regulating and/or time constrained system.

SMS may play a role in determining what object model is available to execute a given simulation. Based on that information, DMS can ensure that the object or event classes being published or subscribed to are legitimate classes within the given object model.

In addition, SMS indicate whether time management is part of a simulation execution. That, in turn, determines whether DMS services related to time management are or are not available.

Requirements

The requirements derived from the DMS/SMS relationship are:

- The composer must have a mechanism to specify the object model that will be part of the simulation execution, and this information must be part of the generated simulation blueprint.
- The composer must have a mechanism that allows the user to specify whether time management will be part of the simulation execution, and this information must be part of the generated simulation blueprint.
- SMS will make object management services available only for the object classes and event classes specified in the object model derived from the simulation blueprint.
- SMS will make time management services available or unavailable based on what the simulation blueprint specifies.
- If a simulation component attempts to publish an object class that does not exist in the object model, the IESM will generate and return an exception.
- If a simulation component attempts to publish an event class that does not exist in the object model, the IESM will generate and return an exception.

- If a simulation component attempts to subscribe to an object class that does not exist in the object model, the IESM will generate and return an exception.
- If a simulation component attempts to subscribe to an event class that does not exist in the object model, the IESM will generate and return an exception.
- If a simulation component attempts to declare itself as time regulating, the IESM will generate and return an exception if time management services are not part of the current simulation execution.
- If a simulation component attempts to declare itself as time constrained, the IESM will generate and return an exception if time management services are not part of the current simulation execution.

A-2: Logical Specification of Declaration Management Services (DMS) and Routing Management Services (RMS) Relationships

Overview of DMS/RMS Relationship

Declaration Management Services (DMS) and Routing Management Services (RMS) are related, because the declarations made by simulation components indicate what routes must be established. For instance, if one simulation component publishes an object class and another simulation component subscribes to that object class, then a route must be established for sending data between those simulation components.

Requirements

The requirements derived from the relationship between these services are:

- When a simulation component declares that it will publish an object class or event class, and other simulation components have subscribed to the same class, the IESM shall ensure that a route exists to transfer data from the publishing component to the subscribing component.
- When a simulation component declares that it will subscribe to an object class or event class, and other simulation components have published the same class, the IESM shall ensure that a route exists to transfer data from the publishing component to the subscribing component.
- When a simulation component declares that it will be time regulating, and other simulation components have declared that they will be time constrained, IESM shall ensure that a route exists between them for sending data that will be queued for receipt at appropriate simulation times. (Note: depending on the implementation, separate routes may not be necessary for timestamped vs. real-time data delivery.)

A-3: Logical Specification of Declaration Management Services (DMS) and Scheduling Management Services (Schemes) Relationships

Overview of DMS/SchMS Relationship

Schedule Management Services (SchMS) control when events are delivered to different systems. Simulation components use Declaration Management Services (DMS) to declare their intent to publish or subscribe to data or to use time management services. The SchMS should give highest priority to the delivery and processing of DMS information. For example, the following may occur in rapid succession immediately after initialization of the simulation:

- Simulation component X declares that it will publish object class A.
- Simulation component Y declares that it will subscribe to object class A.
- Simulation component X declares that it will be time regulating.
- Simulation component Y declares that it will be time constrained.
- Simulation component X creates an object instance of class A.
- Simulation component X updates the object instance using a timestamp in the future.

The IESM must process the declarations (first four bullets above) prior to processing the object creation and update. Otherwise, simulation component Y may not receive the announcement of the object creation if its subscription process has not completed. Furthermore, simulation component X may not be able to schedule future delivery of an update if its declaration to use time management has not been processed, or simulation component Y may receive the object update immediately (rather than at its scheduled simulation time) if the object update is processed prior to simulation component Y's time management declaration.

Requirement

The requirement derived from the DMS/SchMS relationship is:

- The IESM shall guarantee that DMS are processed before Object Management Services (OMS) and Time Management Services (TMS).

A-4: Logical Specification of Declaration Management Services (DMS) and Multi-Level Security Management Services (MMS) Relationships

Overview of DMS/MMS Relationship

Multi-Level Security (MLS) Management Services (MMS) manage data that is exchanged between systems at different classification levels. Typically, systems handle MLS issues by using Cross Domain Solutions (CDS) that allow the secure transfer of electronic data between different security domains. A CDS can be a combination of technology and policy. CDS software implements a specified policy that dictates how data is transferred between systems at different classification levels. The software is typically programmed with rules indicating what data fields can pass freely between security domains, what data must be filtered out, and what data must be modified.

The declarations by simulation components that they will publish and subscribe to different data classes determine where MMS services are required. A CDS must be in place in the following conditions:

- A classified simulation component declares that it will publish an object class or event class that contains classified data.
- Another simulation component at a lower classification level declares that it will subscribe to the same object class or event class.

Requirements

Requirements based on the DMS/MMS relationship are:

- The simulation blueprint generated by the composer should indicate whether MLS is applicable to the simulation execution. (This requirement is repeated in the Object Management Services (OMS)/MMS logical specification.)
- When MLS is applicable to a simulation execution, the simulation blueprint generated by the composer should assign different classification levels to the simulation components.
- When MLS is applicable to a simulation execution, the simulation blueprint generated by the composer should indicate which object classes and event classes can contain classified data.
- In a simulation execution in which MLS is applicable, whenever a simulation component declares that it will publish an object class or event class that contains classified data, and other simulation components at a lower classification level have subscribed to the same class, the IESM shall ensure that a CDS is in place to control data going from the higher-classified component to the lower-classified component.
- In a simulation execution in which MLS is applicable, whenever a simulation component declares that it will subscribe to an object class or event class that contains classified data, and other simulation components at a higher classification level have published the same class, the IESM shall ensure that a CDS is in place to

prevent data going from the higher-classified component to the lower-classified component.

A-5: Logical Specification of Object Management Services (OMS) and Service Management Services (SMS) Relationships

Overview of OMS/SMS Relationship

Service Management Services (SMS) determine which simulation services are accessible to the IESM and to the simulation components. This is important for several reasons:

- The IESM software can operate more efficiently by limiting the services to those essential for executing a given simulation.
- Simulation components can adapt their behavior based on the simulation services available.

Object Management Services (OMS) are essential to any simulation execution, as the IESM serves no purpose if no data is exchanged among simulation components. Thus, an indication that OMS are available is irrelevant; they are always available.

Yet, SMS may play a role in determining what object model is available for a simulation execution. The IESM will be compatible with multiple interoperability standards, including Distributed Interactive Simulation (DIS), Test and Training Enabling Architecture (TENA), and HLA. DIS defines a standard protocol that is, in effect, an object model, whereas TENA includes a set of standardized object models. HLA allows great flexibility for defining any number of object models. If the IESM is to support these standards, then it too must be flexible enough to support any number of object models. The SMS or OMS must provide an appropriate set of core services that the IESM can use to define the permissible objects and events that will be exchanged in a simulation execution, eventually based on the simulation interoperability standard family currently supported. Some of the services offered by SMS may also depend on the existence of certain object classes.

Note: Object management is also affected by whether time management is available for a simulation execution. This is discussed in a separate summary specification on the relationship between Time Management Services (TMS) and SMS.

Requirements

The requirements derived from the OMS/SMS relationship are:

- The composer must have a mechanism that enables the user to specify the object model that will be part of the simulation execution.
- The IESM must have the capability to read the object model specifications for a simulation execution. This includes the permissible object classes and event classes, and the data fields associated with each class.
- The IESM shall ensure that any objects created by simulation components shall specify an object class that exists in the object model defined for the particular execution.

- If a simulation component attempts to create an object class that is not defined in the object model being used for the simulation execution, the IESM shall generate an error notification and send it back to the simulation component.
- The IESM shall ensure that any data fields included in object creations or updates are defined for that object's class.
- If a simulation component attempts to initialize or update a data field in an object instance that is not defined for the associated object class, the IESM shall generate an error notification and send it back to the simulation component.
- The IESM shall ensure that any events sent by simulation components shall specify an event class that exists in the object model defined for the execution.
- If a simulation component attempts to send an event belonging to an event class that does not exist in the object model, the IESM shall generate an error notification and send it back to the component.
- IESM shall ensure that any data fields included in events are defined for the event's class.
- If a simulation component attempts to send an event with data fields not defined for that event class, the IESM shall generate an error notification and send it back to the simulation component.

In addition to the above, if the IESM provides code generation services for exchanging data, these services would have to operate based on a specified object model. For example, automated code generation could produce software that a user could employ to create, update, and delete objects. This auto-generated code could already have safeguards to prevent users from making certain errors, such as including nonexistent data fields.

A-6: Logical Specification of Object Management Services (OMS) and Declaration Management Services (DMS) Relationships

Overview of OMS/DMS Relationship

Object Management Services (OMS) and Declaration Management Services (DMS) are closely related. Using DMS, simulation components declare their intent to publish or subscribe to different categories of object or events. This allows the infrastructure to establish the appropriate routing of data between different simulation components. It is therefore important that simulation components not be permitted to deviate from the declarations when attempting to exchange data with other components.

Requirements

The requirements derived from the relationship between these services are:

- The IESM must permit simulation components to declare their intent to publish object classes and event classes (i.e., declare their intent to send specific categories of data).
- The IESM must permit simulation components to declare their intent to subscribe to object classes and event classes (i.e., declare their intent to receive specific categories of data).
- In DMS, the publications of and subscriptions to object classes and event classes must be limited to the classes defined in the object model being used for the simulation execution.
- The IESM shall generate and return error messages if a simulation component attempts to publish or subscribe to an object class or event class that does not exist in the object model being used.
- In OMS, the IESM shall allow simulation components to create object instances that belong to classes they are publishing.
- In OMS, the IESM shall generate and return error messages if a simulation component attempts to create an object that does not belong to an object class that the component is publishing.
- In OMS, the IESM shall allow simulation components to send events that belong to classes they are publishing.
- In OMS, the IESM shall generate and return error messages if a simulation component attempts to send an event that does not belong to an event class that the component is publishing.
- In DMS, if a simulation component stops publishing an object class that it had previously published, the IESM shall delete any existing objects belonging to that class that the simulation component had previously created.

- In DMS, if a simulation component stops subscribing to an object class to which it had previously subscribed, the IESM shall notify that component to delete any object instances of the object class for which it had previously received object creations.

A-7: Logical Specification of Object Management Services (OMS) and Routing Management Services (RMS) Relationships

Overview of OMS/RMS Relationship

Object Management Services (OMS) are supported by Routing Management Services (RMS), as object and event data must be sent to the appropriate simulation components.

Requirements

The requirements derived from the relationships between these services are:

- When a simulation component creates an object, the IESM shall route the object creation information to all other simulation components that subscribe to the object class of the newly instantiated object.
- When a simulation component updates an object, the IESM shall route the updated object data to all other simulation components that subscribe to the object class of the newly instantiated object.
- When a simulation component sends an event, the IESM shall route the event information to all other simulation components that subscribe to the event class.

Further Requirements

Additional routing considerations apply to data that is published with timestamps. These are described briefly in the logical specification for the TMS and RMS relationship.

A-8: Logical Specification of Object Management Services (OMS) and Schedule Management Services (SchMS) Relationships

Overview of OMS/SchMS Relationship

Schedule Management Services (SchMS) control when events are delivered to different systems. When object and event data is timestamped, the information must be delivered at appropriate simulation times – a topic covered in the "Logical Specification of Time Management Services (TMS) and Schedule Management Services (SchMS) Relationships" specification. Therefore, this specification focuses on scheduling issues not related to time management. For instance, some data transfers must be ordered in a logical way, such as notifying simulation components of an object creation prior to notifying them of an object update. Further, some ordering of non-timestamped object updates and event data may be desirable. For example, it may be useful to deliver data in the same order in which it is sent. However, this may also have runtime performance implications, especially if hundreds of data packets are sent per second.

Requirements

The requirements derived from the OMS/SchMS relationship are:

- When the IESM processes object update data, it should not deliver the data to subscribing simulation components until after it has sent any pending notifications of the object creation of the same object instance to those components.
- When the IESM receives an object deletion, it should deliver any pending updates of the same object instance prior to sending the object deletion notification to subscribing simulation components.
- The simulation blueprint shall allow specification of a Boolean value that dictates that all data sent by a simulation component be delivered to subscribing simulation components in the same order that it was sent.
- The IESM will deliver data to subscribing simulation components in the same order that it was sent if and only if this behavior is indicated in the simulation blueprint.
- Unless otherwise specified, the IESM will send non-timestamped data as fast as possible but will not guarantee that components receive the data in the same order it was sent.

A-9: Logical Specification of Object Management Services (OMS) and Multi-Level Security Management Services (MMS) Relationships

Overview of OMS/MMS Relationship

Multi-Level Security (MLS) Management Services (MMS) manage data that is exchanged between systems at different classification levels. Typically, systems handle MLS issues using Cross Domain Solutions (CDS) that allow the secure transfer of electronic data between different security domains. A CDS can be a combination of technology and policy. CDS software implements a specified policy that dictates how data is transferred between systems at different classification levels. The software is typically programmed with rules indicating what data fields can pass freely between security domains, what data must be filtered out, and what data must be modified. Many CDS products exist in the commercial marketplace, so unless there is a compelling reason for the IESM to develop its own CDS, the IESM should leverage one or more existing products.

As the IESM may be expected to support multiple object models, a different CDS may be required for each object model. Each CDS instantiation could use the same CDS product, but with different rules defined that govern what data fields are discarded or altered when passing data to a system at a lower classification level.

Requirements

Requirements based on the OMS/MMS relationship are:

- The simulation blueprint generated by the composer should indicate whether MLS is applicable to the simulation execution.
- When MLS is applicable to a simulation execution, object data and event data shall not be allowed to pass from a higher classified system to a lower classified system unless it passes through a CDS product that has been approved by the security organization that accredits the overall simulation environment.
- The IESM team shall evaluate existing CDS products to determine their viability for the data exchanges that the IESM is expected to support.
- If the market evaluation identifies viable CDS products, the IESM team shall determine the most effective products for the expected data exchanges.
- If viable CDS solutions exist, the IESM shall integrate one or more of these CDS products.
- If no viable CDS products exist, the IESM team shall develop a CDS driven by rules that specifically define what data passes freely between systems, what data must be filtered out in specific data exchanges, and what data must be altered in specific data exchanges.
- The IESM shall provide a mechanism to associate different object models with different sets of rules for CDS products.

- When data passes through a CDS product, the default behavior for each data field will be to discard the data entirely (i.e., data can only pass through if a rule is defined to allow the transfer).
- The IESM team shall test all CDS products in environments where all systems are at the same classification level (i.e., a lower-classification system would be simulated).
- The IESM will not execute in a true MLS environment until a CDS has been verified and accredited.
- The IESM shall be configurable to allow the logging of which data fields are discarded or altered, so that simulation engineers can quickly show security auditors that the CDS solution is working properly.

A-10: Logical Specification of Time Management Services (TMS) and Service Management Services (SMS) Relationships

Reference

The "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix" summarizes time management services and their potential use in the IESM.[19]

Overview of TMS/SMS Relationship

Service Management Services (SMS) determine which simulation services are accessible. This is important for several reasons:

- The IESM software can operate more efficiently by limiting the services to those essential for executing a given simulation.
- Simulation components can adapt their behavior based on the simulation services available.

Regarding the first point, Time Management Services (TMS) can introduce significant overhead into the infrastructure execution. If TMS will not be used, then the software should not be burdened with any of the time management modules.

Requirements

The requirements derived from the TMS/SMS relationship are:

- The composer must have a mechanism for the user to specify whether time management will be part of the simulation execution.
- The simulation blueprint used to initialize the simulation environment should have a Boolean field to indicate if TMS will be used.
- The IESM must contain a global Boolean variable indicating whether time management services are available; this would be initialized based on the simulation blueprint.
- If the IESM determines that time management is not to be used for an execution, then it will not set up special routes or queues for timestamped data.
- The IESM will have a mechanism that allows a simulation component to query whether TMS are available; when such a query is received, the IESM will respond accordingly.
- If TMS are not accessible for simulation execution, the IESM will generate and return an exception if a simulation component attempts to declare itself as time regulating or time constrained.

- If TMS are not accessible for simulation execution and a simulation component attempts to use a timestamp when sending an object update or an event, IESM will ignore the timestamp and generate an exception indicating that all data exchanges are set to occur in real time.
- If TMS are not accessible for simulation execution and a simulation component makes a request through the application programming interface (API) to advance time, the IESM will generate an exception indicating that all data exchanges are set to occur in real time.

Additional Note

This paper does not include requirements to use different services for conservative and optimistic time management approaches, but these could be considered as well. Use of optimistic time advancement requires the IESM to have the capability to save and restore state in case it becomes necessary to roll back time. This can introduce significant overhead, so the SMS may have to include an indication of whether optimistic time advancement is allowed. The use of optimistic time management is probably a longer-term goal, so the decision as to whether to include this as part of SMS can be deferred.

A-11: Logical Specification of Time Management Services (TMS) and Declaration Management Services (DMS) Relationships

Preface

In the High Level Architecture (HLA), Declaration Management Services (DMS) refer to simulations declaring their intent to send or receive certain categories of information; in other words, they declare what categories of data they will publish and to what categories they will subscribe. By that definition, there is little, if any, overlap between declaration management and time management. However, the IESM does not have to use the HLA paradigm, and this specification assumes that declaration management includes a simulation's declaration of whether and how it will use time management.

Reference

The "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix" summarizes Time Management Services (TMS) and their potential use on IESM.[20]

Declaration Management for Time-Related Issues

As part of declaration management (if one accepts the premise that this includes declaring if and how a participating simulation uses time management), simulations indicate whether they will be time regulating and/or time constrained. The following definitions are used for these characteristics:

- **Time regulating** – If true, this simulation's time can affect the logical time of other simulations, and its outgoing messages go out in timestamp order; if false, this simulation does not have an impact on the logical time of other simulations, and its messages are delivered to other simulations immediately.
- **Time constrained** – If true, this simulation's logical time may be controlled by the time of other time regulating simulations; if false, this simulation's progression is not controlled by other simulations, and any messages sent by other simulations are received immediately.

Time regulating simulations may also declare what time management approach they are using: either **conservative** or **optimistic**.[21] With conservative time management, time regulating simulations are not authorized to advance time until the infrastructure is certain that simulations cannot receive any messages in their past. On the other hand, optimistic time management allows all simulations to advance time beyond the logical time of other simulations, with the expectation that if a simulation receives a message in its past, it will roll back its simulation time as needed.

[20] Tolk, Andreas, "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix," MP180044, The MITRE Corporation, 2018.
[21] Note: This is written with the assumption that the IESM will support optimistic time management. That may not be the case, and even if it does, it will probably be a longer-term goal.

Conservative time management will result in specific requirements. To prevent deadlock, in which no simulation can advance time because all might receive a message in its past, this approach uses a **lookahead** time. This lookahead value defines how far into the future a simulation can produce new messages. Ensuring that at least one of the component simulations has a non-zero value will prevent deadlock.

Requirements

The requirements derived from the TMS/SMS relationship are:

- The IESM must provide a mechanism for a simulation to declare if it is time constrained.
- Each participating simulation is assumed to be not time regulating and not time constrained until it declares otherwise.
- The IESM must allow simulations to declare whether they are time regulating or time constrained at any time, including the capability to change this specification at runtime.
- The IESM must provide a mechanism for a simulation to declare if it is time regulating.
- The IESM must provide a mechanism for a simulation to declare if it is time constrained.
- The IESM must provide a mechanism for a simulation to declare if it is time constrained.
- Each participating simulation shall be assumed to be not time regulating and not time constrained until it declares otherwise.
- The IESM must allow simulations to declare whether they are time regulating or time constrained at any time, including the capability to change this specification at runtime.
- The IESM must provide a mechanism for a time regulating simulation to declare if it supports conservative or optimistic time management.[22]
- By default, the IESM will assume that time regulating simulations use conservative time management.
- For conservative time management, the IESM will provide a mechanism for simulations to specify a lookahead value.

The time management declarations will ultimately affect how messages are exchanged between simulations. These are discussed in other specifications summarized in this appendix, such as those describing the TMS relationship to Object Management Services (OMS) and Routing Management Services (RMS).

[22] Further consideration of the requirement is recommended depending on whether IESM must support both conservative and optimistic time management approaches during the same simulation execution. If not, then all time regulating simulation components must declare the same time management approach.

A-12: Logical Specification of Time Management Services (TMS) and Object Management Services (OMS) Relationships

Reference

The "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix" summarizes Time Management Services (TMS) and their potential use in the IESM.[23] Also, the prior logical specification on the relationship between TMS and Declaration Management Services (DMS) discusses declaration relative to if and how simulation components use time management.

Overview of TMS/OMS Relationship

Object Management Services (OMS) are used to deliver data published by one simulation component to the other simulation components that have subscribed to the data's associated information category. Though this specification refers to it as object management, the concept spans both data on persistent objects and data on ephemeral events. As an example, a simulated aircraft may send updates on its state over time and also generate events, such as a track report; object management pertains to both.

TMS are needed when coupled simulation systems may advance their respective simulation times at different rates. This typically occurs when running a collection of simulations as fast as possible. Without time management, each simulation system would be at a different simulation time, so the systems must use TMS to keep simulation times in synchrony and thereby preserve data integrity.

TMS ensures that the required temporal relationships between simulated objects and events are maintained. For instance, if Simulation A generates a fire event for a missile, Simulation B generates an event for the detection of the missile, and Simulation C generates a third event for the detonation of the missile on its target, then it is important that this data be transmitted in the proper order. Simulation integrity is ruined if a simulation component responsible for defending against the missile receives the detonation event before it has received data on either of the first two events, as now the simulation cannot represent the response to the fired missile. Thus, TMS and OMS must work together to ensure that object and event data generated by different simulation components are delivered to other simulation components at the simulation times that these object updates and events are intended to represent.

TMS is not needed when all simulation systems are set to run at the same rate as wall clock time. In this case, data generated by one simulation can be delivered immediately to other simulations, and any simulation that receives data from an external system can treat it as an update or an event that is occurring at the simulation's current simulation time.

It is also possible that TMS would not be used in any scaled real-time environment in which all simulations are executed in a linear relationship between simulation time and wall clock time. In one approach to scaled real time all simulation components run at a designated

[23] Tolk, Andreas, "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix," MP180044, The MITRE Corporation, 2018.

rate (i.e., a given ratio of simulation time advanced per wall clock time advanced). In this case TMS would not be used, as the individual simulations would be responsible for managing their advancement of simulation time. Other approaches to scaled real time would use TMS. For instance, one simulation component could be the "pacing" simulation that is set to advance time at a specified rate, and the other simulations would not be able to advance their simulation times past the pacer's time. (This is much like a pace car that is seen at car races.)

Requirements

The requirements that address the relationship between TMS and OMS are:

- When time management is not in use, the IESM will attempt to deliver all object and event data generated by a simulation component to all components that subscribe to the associated category of information as fast as possible.
- The IESM must maintain a simulation time for each time regulating or time constrained simulation.
- When a time regulating or time constrained component transmits data on an object or event, it can include an optional time field that can be equal to or greater than its simulation time.
- When time regulating or time constrained components transmit data on objects or events, and the data includes a timestamp earlier than the component's simulation time, the IESM will generate an exception and send it back to the component.
- When a non-time regulating component sends object or event data, the IESM will not associate a timestamp with the data.
- When the IESM processes object or event data that does not have a timestamp, it will prepare it for immediate delivery to all simulation components that subscribe to the associated category of information.
- The IESM will process timestamped object and event data differently for time constrained and non-time constrained subscribers of the data. For time constrained subscribers, the IESM will place the data into timestamp-ordered queues. For non-time constrained subscribers, the IESM will prepare the data for immediate delivery.
- The IESM will provide one or more mechanisms for time regulating or time constrained simulation components to request to advance time. (Note: At first, MITRE suggests a single mechanism for advancing time using a conservative time management approach)
- The IESM will generate and return an exception for a simulation component that requests a time advance to a time that is earlier than its current simulation time plus its lookahead value.
- When a simulation component asks to advance to a specified future time, the IESM will authorize the advance after it has determined that it is impossible for any data

from other simulation components to be delivered to the component with a timestamp earlier than the requested advance time.

- When a time regulating or time constrained component asks to advance to a new time, the IESM will deliver any object or event data in its queue with timestamps less than or equal to the component's new time. This data will be delivered in timestamped order.
- If a time constrained component switches to non-time constrained mode, the IESM will immediately deliver all timestamped data in its queue.

Note: There are many nuances to time management, and when MITRE progresses from logical specifications to more detailed specifications, these requirements must be clarified further.

A-13: Logical Specification of Time Management Services (TMS) and Routing Management Services (RMS) Relationships

Reference

The "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix" summarizes Time Management Services (TMS) and their potential use in the IESM.[24]

Overview of TMS/RMS Relationship

Routing Management Services (RMS) will control, process, and transfer data as specified by an interface specification. Data exchanges between components may vary based on the type of data being transferred.

For simulation components involved in time management, the TMS will coordinate the advancement of simulation time and the delivery of messages in an order that ensures causality is preserved for time-relevant data. When time management is not used, the default behaviors for RMS should not change. However, when timestamped messages are delivered to components, they should be routed to queues from which they will be extracted at appropriate simulation times.

The routing services are designed to have a "not to exceed" data transfer rate that controls how frequently data should be received. Time-managed simulation executions should include the option to turn this feature off, as time management can support cases in which a simulation component does not have to complete its processing of data in a certain time. If a time-managed simulation takes a long time to process data, the overall simulation may run more slowly, but representing the appropriate ordering of events and object updates will preserve the integrity of the data.

Requirements

Though TMS and RMS do not overlap much, the following requirements are derived based on the above discussion:

- The IESM shall provide real-time data transfers between components when time management is not being used.
- Even if a simulation execution has been set up to use time management, data transfers between simulation components shall occur in real time if any of the following conditions is true:
    - The sending simulation component is not time regulating.
    - The receiving component is not time constrained.

[24] Tolk, Andreas, "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix," MP180044, The MITRE Corporation, 2018.

    - The sending simulation's invocation of the service to send an object update or an event does not include the optional timestamp.
- The IESM should route timestamped data (sent from time regulating simulation components to time constrained simulation components) to queues that will be used for future delivery of the data at the appropriate time.
- In time-managed simulation executions, the IESM shall provide an option to turn off the "not to exceed" data transfer rate feature.

A-14: Logical Specification of Time Management Services (TMS) and Schedule Management Services (SchMS) Relationships

Reference

The "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix" summarizes Time Management Services (TMS) and their potential use in the IESM.[25]

Overview of TMS/SchMS Relationship

The Schedule Management Services (SchMS) control when events are delivered to different systems. The obvious correlation with the TMS is that timestamped information must be delivered at the appropriate time to systems that receive the data. This paper does not cover ways to implement this but notes that it can be done in various ways: timestamped data could be queued at the receiver node, queued at some central location, or queued at the sender side.

Requirements

The requirements derived from the TMS/SchMS relationship are listed below. Some of them may be redundant with requirements derived from the logical specifications of TMS with other services.

- Timestamped data will be delivered to time constrained simulation components based on the order of the timestamps, from earliest to latest.
- Timestamped data will be delivered to time constrained components as soon as the receiving component's simulation time is greater than or equal to the timestamp on the data, but not before.
- The IESM will not allow a time constrained simulation component to advance time (i.e., it will not respond to a time advance request) until the IESM is certain that the simulation component will not receive any data with timestamps earlier than its current simulation time.
- After a time regulating simulation component successfully advances time, IESM will calculate the minimum timestamp possible from any of the time regulating simulations in the overall execution. (*Note: In the High-Level Architecture (HLA) this minimum timestamp is referred to as the Lower Bound Time Step, or LBTS.*)
- When the IESM determines that the minimum timestamp possible from any of the time regulating simulations has increased, it will check whether any of the simulation components with pending time requests can be authorized to advance to the previously requested time.

[25] Tolk, Andreas, "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix," MP180044, The MITRE Corporation, 2018.

A-15: Logical Specification of Time Management Services (TMS) and Multi-Level Security Management Services (MMS) Relationships

Reference

The "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix" summarizes Time Management Services (TMS) and their potential use in the IESM.[26] Also, the prior logical specification on the relationship between TMS and Declaration Management Services (DMS) discusses declaration relative to if and how simulation components use time management.

Overview of TMS/MMS Relationship

Multi-Level Security (MLS) Management Services (MMS) manage data that is exchanged between systems at different classification levels. This is typically done using Cross Domain Solutions (CDS) that filter or downgrade data being sent from a classified system to another system at a lower classification level.

For the most part, MMS and associated CDS will not be concerned with time management. However, they must address a few issues. When time management is used, messages being exchanged may have timestamps associated with them. While the inclusion of timestamps would probably not present a security concern, this should be verified, as the timestamps would have to be filtered out if disallowed by security.

As TMS is used to ensure that messages are ordered by time so that causality is represented appropriately, the CDS should preserve the ordering of messages being delivered to a simulation.

Requirements

Requirements based on the TMS/MMS relationship are:

- The IESM must determine if there are any classification issues associated with message timestamps that would be of concern for MLS.
- If the delivery of timestamps to a system at a lower classification level is disallowed, the IESM MMS should filter out the timestamps in the associated CDS.
- MMS and associated CDS must preserve the time-stamped ordering of messages sent to a simulation (even if the timestamps themselves must be stripped out).

[26] Tolk, Andreas, "Time Management Services Issue Paper in Support of Developing the Information Exchange Service Matrix," MP180044, The MITRE Corporation, 2018.

Appendix B: Acronyms

AF	Air Force
API	Application Programming Interface
AWSIM	Air Warfare Simulation
CDS	Cross Domain Solutions
DIS	Distributed Interactive Simulation
DMSO	Defense Modeling and Simulation Office
EAAGLES	Extensible Architecture for Analysis and Generation of Linked Simulations
HLA	High Level Architecture
ICD	Interface Control Document
IESM	Information Exchange Services Matrix
JOPES	Joint Operation Planning and Execution System
JSE	Joint Simulation Environment
JTDS	Joint Training Data Services
LCIM	Levels of Conceptual Interoperability Model
MLS	Multilevel Security
MSCO	Modeling and Simulation Coordination Office
NATO	North Atlantic Treaty Organization
OFP	Operational Flight Program
OMG	Object Management Group
OWL	Web Ontology Language
RPG	Recommended Practice Guide
RTI	Runtime Infrastructure
SMS	Service Management Services
SysML	System Modeling Language
TENA	Test and Training Enabling Architecture
TTCP	The Technical Cooperation Programmed
UML	Unified Modeling Language
VV&A	Verification, Validation and Accreditation