Infrastructure

6

Margaret L. Loper, Tuncer Ören, Cláudio Gomes, Valdemar Vicente Graciano Neto, and Ernest H. Page

Abstract

Providing the appropriate infrastructure for simulation is the topic of this chapter of the SCS M&S Body of Knowledge. It provides sections on various simulation standards, standard organizations, and compliance certificates. Publications of some applicable codes of best practices and lessons learned are provided as well as a section on resource repositories. The currently dominant standard Distributed Interactive Simulation (DIS) and High-Level Architecture (HLA) conclude the chapter.

Keywords

Modeling and simulation · Simulation standards · Simulation standard organizations · Distributed interactive simulation (DIS) · High-level architecture (HLA)

M. L. Loper (✉)
Georgia Tech Research Institute, Atlanta, GA, USA
e-mail: Margaret.Loper@gtri.gatech.edu

T. Ören
University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

C. Gomes
Aarhus University, Aarhus, Denmark
e-mail: claudio.gomes@ece.au.dk

V. V. Graciano Neto
Federal University of Goiás, Goiania, Brazil

E. H. Page
The MITRE Corporation, McLean, VA, USA
e-mail: epage@mitre.org

T. Ören et al. (eds.), *Body of Knowledge for Modeling and Simulation*,
Simulation Foundations, Methods and Applications,
https://doi.org/10.1007/978-3-031-11085-6_6

6.1 Standards

Margaret L. Loper

Standardization involves the use of common products, processes, procedures, and policies to facilitate attainment of business objectives. Standardization is about enabling interoperability: a fundamental objective of all stakeholders, be they policy-makers, industrial players, or users. Numerous industrial initiatives in a variety of different economic sectors owe their success to a commitment of the stakeholders to join forces to agree on open specifications for interoperable systems. Since the earliest days of distributed simulation, standards have played a crucial role in achieving interoperability.

The most widely used distributed simulation standards in place today are the Distributed Interactive Simulation (DIS) Protocol, the High Level Architecture (HLA), and the Test- and Training Enabling Architecture (TENA). There are various means to establish standards, and the communities responsible for these Live, Virtual, and Constructive (LVC) simulation standards have chosen different approaches.

6.1.1 De Jure, De Facto, and Proprietary Standards

There are three basic types of standards in existence today and prevalent in the IT industry:

- De Jure standard: endorsed by a standards organization (TechEncyclopedia http://www.techweb.com/encyclopedia/defineterm.jhtml?term=de+jure+standard);
- De Facto standard: widely used, but not endorsed by a standards organization (TechEncyclopedia http://www.techweb.com/encyclopedia/defineterm.jhtml?term=defactostandard); or
- Proprietary standard: belongs to an entity that exercises control over the standard.

The three types of standards are not orthogonal. There are cases where the lines between the types of standards may become blurred or combined. An example of blurring the lines between De Facto and proprietary standards is the two “standards” for High Definition DVD formats. Each standard is supported by a group of vendors, and the formats are incompatible. The expectation is that one of the proprietary standards will become the community De Facto standard for digital video recording, much like the battle some years ago between VHS and BETA formats. An example of combining types of standards is the BMD benchmark environment used by the Missile Defense Agency (MDA). The MDA simulation community has created an environment for its developers to benchmark new algorithms and components. The environment, considered an MDA standard, is based on the proprietary MATLAB environment. Thus, MDA has created De Facto standards which use proprietary standards as its foundation.

6.1.2 Open Standards

Open standard is another term often used when discussing standards. An open standard is more than just a specification; the *principles* behind the standard and the *practice* of offering and operating the standard are what makes the standard Open. The term "open standard" may be seen from perspectives of its stakeholders ("Open Standards Requirements", Ken Krechmer, http://www.csrstds.com/openstds.pdf:

- Organizations representing the standards creators consider a standard to be open if the creation of the standard follows the tenets of open meeting, consensus, and due process.
- An ***implementer*** of a standard would call the standard open when it serves the market they wish, it is without cost to them, does not preclude further innovation (by them), does not obsolete their prior implementations, and does not favor a competitor.
- The ***user*** of an implementation of the standard would call a standard open when multiple implementations of the standard from different sources are available, when the implementation functions in all locations needed, when the implementation is supported over the user-planned service life, and when new implementations desired by the user are backward compatible to previously purchased implementations.

There are numerous definitions of an open standard by national standards bodies (http://en.wikipedia.org/wiki/Open_standard). The definition by Krechmer lists ten requirements that enable open standards.

6.1.3 Standards Organizations

A standards organization is any entity whose primary activity is developing, coordinating, promulgating, revising, amending, reissuing, interpreting, or otherwise maintaining standards that address the interests of a wide base of users. There are two general types of standards organizations: standards developing organizations (SDO) and standards setting organizations (SSO).

SDOs are formal organizations accredited to develop standards using open and transparent processes. Examples include the International Organization for Standardization (ISO) and the Institute of Electrical and Electronics Engineers (IEEE). SSOs refer to organizations that set what the market perceives as standards. The term "recognized SSO" refers to any SSO recognized directly or indirectly by a government entity. Consortia is the term used for SSOs that are not recognized SSOs. Examples of a "recognized SSO" include the World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF).

6.1.4 M&S Standards Organizations

M&S standards organizations can be classified into two types: government and commercial. Government refers to standards forums under US government control. These types of standards organizations are typically composed of systems engineers and technical leads of major DoD stakeholders of the architecture. They discuss requirements, design trade-offs, and issues associated with the architecture. These standards organizations also have contractor support that is responsible for architecture design and prototyping. Simulation-related standards that have been created using this approach include TENA.

Commercial refers to standards created in open forums outside of government control. Examples of this include IEEE, SISO, International Organization for Standardization (ISO), and Object Management Group (OMG). These types of organizations are composed of users, vendors, academics, government organizations, and developers of the architecture. Like government forums, they discuss requirements, trade-offs, and other issues associated with the architecture. However, they do not have contractor support for architecture design and prototyping. Instead, these forums rely on members to develop prototypes and provide technical feedback on the architecture specifications.

Another model of standards development that has been successfully used for LVC architectures is a combination of government and commercial organizations. This was demonstrated with the first set of HLA standards. The government was responsible for developing and evolving the early versions of the HLA specifications. This enabled DoD stakeholders to include requirements and provide technical feedback resulting from their programs. Once they reached a point of maturity, the HLA specifications were transferred to SISO and went through IEEE standardization. The HLA standards were also taken to OMG to be standardized. Similarly, the Synthetic Environment Data Representation and Interchange Specification (SEDRIS) (http://www.sedris.org/ and http://en.wikipedia.org/wiki/Sedris) standards were initially developed as government standards and then taken to ISO for standardization. Using IEEE, OMG, and ISO enabled the standards to receive a broader commercial review. Simulation-related standards that have been created using this approach include DIS, HLA, and SEDRIS.

There are two main standards developing organizations in the LVC community today: the Architecture Management Team, which develops TENA standards, and SISO, which develops DIS and HLA standards. In addition to these standards organizations, the DoD services each have a group responsible for coordinating standards use, both from developing object model content (i.e., FOMs) as well as endorsing standards that meet the requirements of their programs. These groups have people that participate in the AMT or SISO, but they do not have formal representation nor formal requirements generation functions for these standards developing bodies.

There are also commercial standards organizations involved in developing specifications and standards for technologies related to LVC. For example, the Internet Engineering Task Force (IETF) develops communication standards, including security; the World Wide Web Consortium (W3C) develops web-related standards such as SOAP and XML; the OMG develops modeling standards such as UML, SysML, and UPDM; OASIS and the Open Group have developed specifications for the service-oriented architecture; and ISO has standardized SEDRIS. Thus, there is a hybrid approach to standards, encompassing standards and technologies from all IT-related organizations. However, there is little, if any, coordination among these standards development activities resulting in a stovepipe approach to standards management.

6.1.5 Compliance Certification

The overarching purpose of compliance certification to a standard is to ensure that products adhere to that particular standard. Compliance certification provides a level of security to users of compliant products and provides a level of assurance that certified products satisfy some set of requirements. Compliance certification is an important element of the standards process.

Compliance certification may be defined as the act or process of determining compliance to a defined standard. The primary reasons for standards compliance in the M&S LVC domain are a greater probability of interoperability between simulation assets and a greater probability for reuse of those assets in different configurations. A number of processes are in use today with existing LVC standards. Those processes range from very informal approaches such as checklists to formal compliance tests. Operational certification is most often associated with verification and validation however.

6.2 Code of Best Practice

Tuncer Ören.

"The set of best practices recommended for use for any MS&A application includes:

1. conceptual modeling practice,
2. innovative approaches,
3. software engineering practice,
4. model confidence/ verification, validation, and accreditation (VV&A),
5. use of standards,

6. interoperability,
7. execution performance, and
8. user-friendliness and accessibility" [1].
9. country modeling [2],
10. military and business [3, 4],
11. networks [5],
12. participatory modeling [6], and
13. railways [7].

Code of best practices for: Crash modeling and simulation [8], engineering applications [9, 10], healthcare [11, 12], homeland security applications [1], and modeling and simulation [13, 14].

6.3 Lessons Learned

Tuncer Ören.

The following list comprises seminal papers comprising lessons learned from selected application domains.

6.4 Resource Repositories

Valdemar V. Graciano Neto and Cláudio Gomes.

The multiplicity of Modeling and Simulation (M&S) formalisms and simulation paradigms is high. That diversity often forces researchers to produce their own simulation models from scratch every time they initiate a new project due to difficulties in reusing existing models. Those difficulties range between (i) not knowing whether similar models already exist, (ii) differences in formalisms, even when models were produced for a similar domain, and (iii) lack of documentation about how to use such models. The Modeling and Simulation Resources Repositories (MSRR), also known as resource libraries or suites, have potential to foster reuse by gathering a diversity of resources in a unified access point. Resources (also known as assets or artifacts) such as models (simulatable or not-simulatable [15]), experimental frames, pairs of base and lumped models [16], specifications of physical environments and scenarios, datasets, composable simulation components and simulation services can be made available to a large audience [17]. Models capturing specific domains can be available in several formats, such as XML, UML, or DEVS, and model transformations can also be available to transform non-executable models into simulatable formats.

Sharing and exchanging models have the potential to accelerate the systems development. Efforts have been made, for instance, to standardize the representation of physical system models, through languages such as Modelica [18], and interfaces such as the Functional Mockup Interface (FMI) standard [19]. Modelica standard library is a collection of modular physical system models and common block diagram elements enabled by the Modelica Language, while FMU Cross Check Repository is a collection of black box simulators exported from different modeling and simulation tools. The Modelica standard library allows a researcher to quickly create a model for a physical system by reusing pre-existing components in the standard library. The FMI standard, in turn, through its black box and Intellectual Property (IP) protecting interface, enables an unprecedented level of integration of models (as black boxes) provided by different and even competing suppliers.

These advances democratize M&S by making it cheaper to produce high-quality models of the system, which in turn can be more easily exchanged with researchers. Smaller companies and universities can then reap the benefits of M&S, speeding up innovation. Another advantage brought by resources repositories is that they enable benchmarking. For instance, researchers who create a new machine learning technique can apply it to many freely available datasets. Over time, benchmarks emerge when researchers are expected to tackle their contributions. This leads to more mature contributions and easy comparison with existing ones.

Ideally, a resource repository should be capable of: Catalog/index/organize the resources stored, persist, and allow for resources search and retrieve, resource management, and resources delivery through well-defined interfaces, resource stores (analogous to application stores), and as services (simulation as a service, for instance) [20]. Resources repositories are common in other areas, such as software engineering [21, 22] and biology [23]. MSRR have also been proposed over the years [24–26] (https://ntrs.nasa.gov/citations/20060023324).

However, several challenges still remain, such as a standardized representation in order to enable their existence. This is hard to be achieved due to the diversity of formalisms, which can be categorized as [27–31]:

- Time Domain—The time can be a singleton (e.g., algebraic equations), a continuous set (e.g., Ordinary and Algebraic Differential Equations), discrete set (e.g., Difference equations, Petri-nets, Automata), or superdense set (e.g., Hybrid Automata and Classic DEVS). In Superdense time [32–34], each time point is a pair consisting of a real number and a natural.
- State Domain—The state domain can be a continuous set (e.g., ODEs and DAEs), a discrete set (e.g., Petri-nets), or a mix of both (e.g., DEVS and Hybrid Automata).
- Behavior Trace—The behavior trace can be discontinuous (e.g., DEVS and Hybrid Automata), and continuous (e.g., ODEs and DAEs).
- Causality—Models can be a-causal, when they can be coupled to other models without any notion of inputs and outputs (e.g., DAEs), or causal, when outputs need to be connected to inputs and vice-versa (e.g., DEVS and Difference Equations).

- Evolution—The evolution of the state can be deterministic (e.g., DEVS), stochastic (e.g., Markov Chains), or non-deterministic (e.g., Hybrid Automata). To overcome these difficulties, the following suggestions have been made in the literature to conform different formalisms to make them be coupled [35]:
- Super Formalism—The formalisms used to express each model are unified into one single formalism, with well-defined syntax and semantics. This is what is done in [36, 37]. Other examples include: timed Petri-nets, Markov chains, etc.
- Common Formalism Reduction—The models are transformed into a model that is expressed in a single formalism. The "common" adjective refers to the fact that each model can be transformed into a restricted set of formalisms. Hence, one formalism must be found to which all models to be integrated can be transformed. For example, differential equations can be used to represent the model of a PID controller sampling a plant model. The latter was originally modeled as a differential equation. More examples are detailed in [35].

Co-simulation can be seen as taking the common formalism reduction integration technique to the extreme, where models that produce behavior are coupled solely on their behavior (inputs/outputs, over time). However, automatically configuring co-simulation can be very difficult [29, 38].

For more challenges and potential solutions to establishing model repositories, we recommend the following references. Basciani et al. [23] established a discussion on the reality of resource repositories some years ago. Zeigler et al. [20] show how to build a model suite relying on the MS4 Me tool and Ören [17] presents requirements necessary to achieve reuse through MSRR.

6.5 Distributed Interactive Simulation (DIS)

Ernest H. Page, Margaret L. Loper.

For nearly a half-century, the defense simulation community has explored, developed, and applied technologies and methods that support the runtime interoperation of simulations and other systems. Major milestones in this history are:

- DARPA Simulator Networking (SIMNET) program [39–42]
- Distributed Interactive Simulation (DIS) protocol [43–47]
- Aggregate Level Simulation Protocol (ALSP) [48, 49]
- High Level Architecture (HLA) for Modeling and Simulation [50], and
- Test and Training Enabling Architecture (TENA) [51]

Within this M&SBoK, we highlight DIS and HLA. But a brief discussion of SIMNET is warranted to establish the context for each of these distributed simulation standards.

6.5.1 Simnet

Shortly after the development of the ARPANET, DARPA initiated the SIMNET program to investigate the feasibility of using networked simulators to support group training (also referred to as *collective* training) at large scales and at great distances. The SIMNET vision was a large-scale, interactive, collection of networked simulations forming a *synthetic environment* that could be entered by any authorized combatant from anywhere on the network using his/her simulator as a porting device. The initial project scope called for a simulator networking testbed with four geographically distributed sites hosting 50–100 vehicle simulators each, with a focus on slower-moving ground-based platforms, e.g., tanks and armored personnel carriers. The project required technological advances in a variety of areas, including image generation, distributed databases, and real-time network protocols. Key design principles for SIMNET included:

- *Selective fidelity*. In order to minimize simulator costs, a simulator should only contain high fidelity representations of those elements essential to the training task. Everything else should be represented at lower fidelities, or not all.
- *Autonomous simulation nodes*. Each node is responsible for maintaining the state of at least one object in the synthetic environment, and for communicating to other nodes any events caused by its object(s). Each node receives event reports from other nodes and calculates the effects of those events on its objects. All events are broadcast on the simulation network and are available to any node that is interested. There is no centralized controlling process. Nodes may join and leave the network without affecting other nodes. Each node advances simulation time according to a local clock.
- *Transmission of ground truth data*. Each node transmits the absolute truth about the current state of the object(s) it represents. Alteration of data to suit simulation objectives is the responsibility of the receiving node. For example, the position of a vehicle is broadcast to the network with 100% accuracy. If an object in another simulator determines that it would perceive the vehicle through a particular sensor, with an accuracy determined by the alignment of the sensor and current weather conditions, then the receiving simulator should degrade the reported position accordingly.
- *Transmission of state change information*. To minimize network loading, nodes transmit state update information only. To accommodate late-joining nodes and networks with high packet loss, this rule is often relaxed. In these situations, nodes send periodic (but relatively infrequent) updates for each owned object regardless of whether or not their state changes. This update interval is known as the "heartbeat."
- *Dead reckoning*. Between state update messages, receiving nodes may extrapolate the last reported state of remote objects that are of interest. To keep the extrapolated values and actual values roughly aligned, the sending node maintains the same approximation used by the receiving node(s) and transmits a state

update whenever the true position (or orientation) of an object diverges from the calculated dead reckoned values by more than an agreed-upon threshold. Lin [52] and Fujimoto [53] discuss common dead reckoning algorithms.

SIMNET was adopted by the Army as the basis for the Combined Arms Tactical Trainer (CATT) in 1990 and continued to be used in a variety of programs until supplanted by the DIS standard. SIMNET has been identified as one of the most significant transitions of technology from DARPA to DoD [40].

6.5.2 Origins of the DIS Protocol

Recognizing the importance of the SIMNET program and concerned that activity related to networked simulation was occurring in isolation, a small conference was held in April 1989 called "Interactive Networked Simulation for Training." The group believed that if there were a means to exchange information between companies, distributed simulation technology would advance more rapidly. The group also believed that technology had stabilized enough to begin standardization. The conference developed into the Distributed Interactive Simulation (DIS) Workshops.

Through these workshops, networked simulation technology and the consensus of the community were captured in proceedings and standards. The standards initially focused on SIMNET, but evolved to include a broader range of technology areas. DIS Workshops were held semi-annually from 1989 through 1996. In 1996, the DIS Workshops transformed itself into a more functional organization called the Simulation Interoperability Standards Organization (SISO), which focused on creating standards for the broad area of simulation interoperability. The first Simulation Interoperability Workshop (SIW) held under the SISO banner was the 1997 Spring SIW in Orlando. SIWs have continued since 1997, holding some workshops at various locations in Europe.

The Distributed Interactive Simulation (DIS) protocols became the Institute of Electrical and Electronics Engineers (IEEE)1278.1 standard in 1993. The fundamental design principles for DIS follow directly from SIMNET, and much of the standardization effort focused on extending the basic SIMNET communication structure—the Protocol Data Unit (PDU)—a bit-encoded packet for communicating entity state and other types of information necessary for distributed combat simulations, e.g., weapons fire and weapons detonation events.

Like SIMNET, DIS was designed to support the internetworking of simulations that run in real-time. Whereas SIMNET had achieved the ability to support relatively small numbers of concurrently running simulators representing platoon and squad-sized engagements, the vision for DIS was to support the interoperation of thousands of simulators/simulations and scale to a military campaign level (tens to hundreds of thousands of battlefield entities). This appetite for scale led to a burgeoning market in Semi-Automated Forces (SAF). SAFs—a concept initiated within SIMNET—were used to populate synthetic environments with background objects that behaved in a "reasonable" way [46]. They were dubbed "semi-automated" because human intervention was often required to make the modeled entities maintain their reasonable behavior. However, the power and utility of SAFs were recognized very quickly. Entity behavior in SAFs became the focus of numerous conferences, workshops, and texts. SAFs were a ripe area for research in Artificial Intelligence engines such as Soar [54]. DIS-supported simulation environments consisting entirely of SAFs became commonplace.

One of the lasting contributions of the DIS Workshops was the definition of Live, Virtual, and Constructive (LVC) simulations. This taxonomy categorizes simulations by the way in which humans interact with them. Live simulation refers to real people operating real systems (e.g., a pilot flying a jet) for a simulated purpose. A virtual simulation is one that involves real people operating simulated systems (e.g., a pilot flying a simulated jet). Constructive simulations are those that involve simulated people operating simulated systems (e.g., a simulated pilot flying a simulated jet).

6.5.3 DIS Today

The goal of DIS is to create a common, consistent simulated world where different types of simulators can interact. Central to achieving this goal is a set of IEEE standards. The most commonly used standard is 1278.1, which describes the PDUSs. The first DIS standard defined 10 PDUs; the most recent standard, DIS 7, was published in 2012 and defines 72 PDUs arranged into 13 families. The approved IEEE Standards for DIS include:

- IEEE 1278.1—Application Protocols
- IEEE 1278.1A—Enumeration and Bit-encoded Values
- IEEE 1278.2—Communication Services and Profiles
- IEEE 1278.3—Exercise Management & Feedback (EMF)
- IEEE 1278.4—Verification Validation and Accreditation
- IEEE P1278.5—XXXX—Fidelity Description Requirements (never published).

In addition to the IEEE standards, SISO maintains and publishes an "enumerations and bit-encoded fields" document yearly. This document is referenced by the IEEE standards and used by DIS, TENA, and HLA.

From an implementation perspective, simulation owners either custom-develop DIS interfaces or buy commercial products. There is also an open-source initiative, Open-DIS, to provide a full implementation of the DIS protocols in C++ and Java [55].

There have been numerous DIS federation events over the last 25 years. Two examples are "bookend" LVC events presented at the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC). Twenty-three years spanned the two events, and while technology has progressed, some aspects have not progressed as quickly as we might think. The 1992 event was the first-ever demonstration of DIS and distributed simulation among dissimilar, heterogeneous simulations [45]. The 2015 event was an effort to recreate the demonstration with modern technology and architectures [56].

6.6 High Level Architecture (HLA)

Ernest H. Page, Margaret L. Loper.

By 1995, the evidence was clear that interconnecting simulations could be of practical value. SIMNET provided an efficient and effective mechanism for linking man-in-the-loop simulators. DIS extended SIMNET and provided scalability to many thousands of entities in SAF-based exercises. Another DARPA project, the Aggregate Level Simulation Protocol (ALSP), developed a capability to interconnect "logical time," e.g., discrete event, simulations [49]. Also by this time, many defense simulations had interconnection interfaces—some SIMNET, some DIS, some ALSP, some "homegrown," and some had multiple such interfaces. To mitigate against the proliferation of interconnection approaches, the DoD, through the Defense Modeling and Simulation Office (DMSO) and SISO, began developing a standard for simulation interconnection known as the High Level Architecture (HLA). The HLA was envisioned as an approach to bridge live, virtual, and constructive simulations in one architecture, representing a generalization and extension of SIMNET, DIS, and ALSP. The HLA architecture is defined by three components:

- An Object Model Template—a common model definition and specification formalism,
- An Interface Specification—a collection of services describing the HLA runtime environment, and
- The HLA Rules—governing compliance with the architecture.

The HLA standards began in 1995 under a government standards process managed by DMSO. The DoD adopted the baseline HLA architecture in 1996 and the standards were moved to an open standards process managed by SISO. The IEEE standards for HLA, first approved in 2000 and updated in 2010, include:

- 1516—Framework and Rules
- 1516.1—Federate Interface Specification
- 1516.2—Object Model Template (OMT) Specification
- 1516.3—Federation Development and Execution Process (FEDEP) Recommended Practice
- 1516.4—Recommended Practice for Verification, Validation, and Accreditation of a FederationAn Overlay to the HLA FEDEP

The HLA was conceived to have applicability across the full range of defense simulation applications, including those used to support training, mission rehearsal, analysis, and test and evaluation.

At core of HLA is the notion of a *federation*. A federation is a collection of federates—simulations and other systems—that interoperate using the protocols described by the architecture. The HLA is based on the idea of separating the functionality of simulations from the infrastructure required for communication among simulations. This separation is accomplished by a distributed operating system called the Run-Time Infrastructure (RTI). The RTI provides common services to simulation systems and provides efficient communications to logical groups of federates. Federation execution is accomplished through the RTI, which is an implementation of the services defined by the interface specification.

In contrast to SIMNET and DIS, HLA includes time management services to support event ordering [57]. Both time stamp order, where messages are delivered to simulations in order of time stamp, and receive order, where messages are delivered to simulations in order received, are supported in HLA. While HLA provides global time management, use of these services is not required. Simulations can choose to advance time at its own pace, not synchronized with other simulations.

In contrast to the static DIS PDUs, HLA uses the concept of OMTs to specify the information communicated between simulations. This enables users to customize the types of information communicated among federates based on the needs of the federation. A Federation Object Model (FOM), and instantiation of the OMT, provides the model specification and establishes a contract between the federates with respect to the nature of the activity taking place during federation runtime.

In a typical federation execution, a federate joins the federation, indicates its operating parameters (e.g., information the federate will provide to the federation and information it will accept from the federation), and then participates in the evolution of federation state until the federate departs the federation, or the simulation terminates. FOM data is provided to the RTI at runtime, enabling the infrastructure to enforce the information contract that the FOM represents.

In 1996, HLA compliance was mandated for all defense simulations, with the intention that support for other protocols would cease [58]. To accommodate DIS applications the Real-time Platform Reference (RPR), FOM was developed which defines a translation between DIS PDUs and HLA services [59]. As with an earlier mandate of the programming language Ada, however, the “No Can Pay/No Can Play” HLA mandate was perceived as onerous and became too unwieldy to enforce.

Distributed simulation architectures are designed to meet the needs of one or more user communities, and the design choices made by the HLA attempted to improve on perceived shortcomings of existing architectures [60]. The static nature of DIS PDU’s was identified as a significant problem; as the real world is always changing. A flexible object model capable of modeling changing data without having to continuously change the underlying standard was seen as a better approach. Allowing users to define their data exchange based on specific requirements using the OMT was seen as providing improved object model extensibility.

However, increased flexibility to the user also allowed users to independently develop a plethora of object models that were rarely interoperable. Additionally, HLA adopted an API Standard as opposed to an on-the-wire standard that allowed it to more rapidly adopt technological advancements in how data are transmitted. While this enabled commercial RTI developers the freedom to innovate and optimize their RTI implementations, the result was non-interoperable RTIs. In practice, when disparate RTI versions are used in a given event, gateways or other inter-protocol translation mechanisms are used to bridge the federates.

Today, both HLA-compliant and DIS-compliant simulations abound. Since HLA separates the functionality of simulations from the infrastructure, it has had more success in being adopted by non-DoD applications, including NASA, transportation, and supply chain management. The existence of multiple architectures means users will select the methodology that best meets their needs. This often results in multiple architectures being used in the same federation execution. In this case, incompatibilities between DIS, HLA, and TENA require the development of point solutions to effectively integrate the various architectures into a single, unified set of simulation services. The future of distributed simulation to solve and understand complex problems will rely on the development of simulation standards.

References

1. Jain S, McLean CR (2011) Best practices for modeling, simulation and analysis (MS&A) for homeland security applications. U.S. Dept. of Commerce, National Institute of Standards and Technology. NISTIR 7655. https://www.govinfo.gov/content/pkg/GOVPUB-C13-ae22d29a1eaa5577f0dc3f67b51bc9ba/pdf/GOVPUB-C13-ae22d29a1eaa5577f0dc3f67b51bc9ba.pdf
2. Silverman B et al (2021) StateSim: lessons learned from 20 years of a country modeling and simulation toolset. Social computing, behavioral-cultural modeling and prediction and behavior representation in modeling and simulation annual conference. https://repository.upenn.edu/cgi/viewcontent.cgi?article=1939&context=ese_papers
3. Barth R et al (2021) typical pitfalls of simulation modeling - lessons learned from armed forces and business. J Artific Soc Soc Simul 15(2). https://doi.org/10.18564/jasss.1935
4. Hofmann MA (2004) Criteria for decomposing systems into components in modeling and simulation: lessons learned with military. https://doi.org/10.1177/0037549704049876
5. Anton SD (2018) The dos and don'ts of industrial network simulation: a field report. ISCSIC '18: proceedings of the 2nd international symposium on computer science and intelligent control, September 2018 Article No.: 6, pp 1–8 https://doi.org/10.1145/3284557.3284716
6. Sterling EJ et al (2019) Try, try again: Lessons learned from success and failure in participatory modeling. https://doi.org/10.1525/elementa.347
7. Meijer S (2012) Gaming simulations for railways: lessons learned from modeling six games for the dutch infrastructure management. https://doi.org/10.5772/35864 https://www.intechopen.com/books/infrastructure-design-signalling-and-security-in-railway/gaming-simulations-for-railways-lessons-learned-from-modeling-six-games-for-the-dutch-infrastructure
8. Fasanella EL et al (2002) Best practices for crash modeling and simulation. NASA/TM-2002–211944, ARL-TR-2849. https://ntrs.nasa.gov/api/citations/20020085101/downloads/20020085101.pdf
9. Fasanella EL, Jackson KE (2002) Best practices for crash modeling and Simulation. NASA/TM-2002–211944, ARL-TR-2849. https://www.researchgate.net/publication/24302082_Best_Practices_for_Crash_Modeling_and_Simulation

10. Mare JC (2019) Best practices for model-based and simulation-aided engineering of power transmission and motion control systems. Chin J Aeronaut 32(1):186–199. https://www.sciencedirect.com/science/article/pii/S1000936118302668
11. Erdemir A et al (2020) Credible practice of modeling and simulation in healthcare: ten rules from a multidisciplinary perspective. J Transl Med 18:369. https://translational-medicine.biomedcentral.com/track/pdf/https://doi.org/10.1186/s12967-020-02540-4.pdf
12. Standards of Best Practices: Simulation. INACSL (The International Nursing Association for Clinical Simulation and Learning). https://www.inacsl.org/inacsl-standards-of-best-practice-simulation/
13. JHU-APL (2010). Best Practices for the Development of Models and Simulations. Final Report, NSAD-L-2010–129, JNC04. https://acqnotes.com/Attachments/JHU%20APL%20Best%20Practices%20for%20the%20Development%20of%20M&A,%20June%2010.pdf
14. Morse KL (2010) Models and simulations development best practices. In: 13th annual NDIA systems engineering conference. https://ndiastorage.blob.core.usgovcloudapi.net/ndia/2010/systemengr/WednesdayTrack5_10735Morse.pdf
15. Zeigler B, Sarjoughian H (2017) Model development and execution process with repositories, validation, and verification. In: Guide to modeling and simulation of systems of systems, Springer
16. Zeigler BP (2019) Simulation-based evaluation of morphisms for model library organization. In: Model engineering for simulation, Academic Press
17. Ören T (2005) Toward the body of knowledge of modeling and simulation. In: Interservice/industry training, simulation, and education conference (I/ITSEC), NTSA
18. Tiller M (2012) Introduction to physical modeling with modelica, vol 615. Springer
19. Blochwitz T et al (2012) Functional mockup interface 2.0: the standard for tool independent exchange of simulation models. In: Proceedings of the 9th international modelica conference. Linköping University Electronic Press.
20. Zeigler BP et al (2013) Creating suites of models with system entity structure: global warming example. In: Wainer GA, Mosterman PJ, Barros FJ, Zacharewicz G (eds) Proceedings of the spring simulation conference, IEEE
21. France RB, Bieman JM, Mandalaparty SP, Cheng, Betty HC, Jensen AC (2012) Repository for model driven development (ReMoDD). In: Glinz M, Murphy GC, Pezzè M (eds) Proceedings of the international conference on software engineering, IEEE.
22. Koegel M, Helming J (2010) EMFStore: a model repository for EMF models. In: Kramer J, Bishop J, Devanbu PT, Uchitel S (eds) Proceedings of the 32nd ACM/IEEE international conference on software engineering, vol 2. IEEE.
23. Basciani F et al. (2015) Model repositories: Will they become reality? Proceedings of the International Workshop on Model-Driven Engineering on and for the Cloud (CloudMDE), Richard Paige, Jordi Cabot, Marco Brambilla, James H. Hill (Eds.), CEUR.
24. Liu HB, Su HY, Chai XD, Zhang H, Zhan SY (2009) Research and application of XML in modeling and simulation resource repository. J Syst Simul 22
25. Milroy A, Hale J (2006) Modeling and simulation resource repository (MSRR) (System Engineering/integrated M&S Management Approach), Audrey Milroy, QTEC, Inc. and Joe Hale, NASA MSFC
26. Yan B, Yang Z (2009) Simulation resource selection based on local optimization method in distributed simulation repository. In: Proceedings of the 16th international conference on industrial engineering and engineering management, IEEE
27. Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H (2017) Co-simulation: state of the art. University of Antwerp, February 1
28. Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H (2018) Co-simulation: a survey. ACM Comput Surv 51(3):49:1–49:33
29. Gomes C (2019) Property preservation in co-simulation. University of Antwerp. https://repository.uantwerpen.be/docman/irua/57f437/163840.pdf

30. Jantsch A, Sander I (2005) Models of computation and languages for embedded system design. IEEE Proc Comput Digital Tech 152(2), 15: 114–129, IEEE
31. Vangheluwe H (2008) Foundations of modelling and simulation of complex systems. Electronic Communications of the EASST 10
32. Lee EA, Zheng H (2005) Operational semantics of hybrid systems. In: Morari M, Thiele L (eds) Hybrid systems: computation and control. Lecture Notes in Computer Science, vol 3414. Springer, pp 25–53
33. Maler O, Manna Z, Pnueli A (1992) From timed to hybrid systems. In: de Bakker JW, Huizing C, de Roever WP, Rozenberg G (eds) Real-time: theory in practice, lecture notes in computer science. Springer
34. Manna Z, Pnueli A (1993) Verifying hynrid systems. In: Clarke E, Henzinger T, Veith H, Bloem R (eds) Handbook of model checking. Lecture Notes in Computer Science. Springer
35. Vangheluwe H, De Lara J, Mosterman PJ (2002) An introduction to multi-paradigm modelling and simulation. In: Zeigler B, Barros F (eds) Proceedings of the AI, simulation and planning in high autonomy systems conference, IEEE
36. Mustafiz S et al (20216) Towards modular language design using language fragments: the hybrid systems case study. In: Latifi S (ed) Proceedings of the 13th international conference on information technology—new generations (ITNG), Springer
37. Mustafiz S et al (2016a) Modular design of hybrid languages by explicit modeling of semantic adaptation. In: Barros F, Prähofer H, Hu X, Denil J (eds) Proceedings of the symposium on theory of modeling and simulation: DEVS integrative M&S symposium, SCS
38. Oakes BJ, Gomes C, Holzinger FR, Benedikt M, Denil J, Vangheluwe H (2021) Hint-based configuration of co-simulations with algebraic loops. In: De Rango F, Wagner G, Werner F, Ören T (eds) Simulation and modeling methodologies, technologies and applications, advances in intelligent systems and computing, Springer
39. Cosby L (1995) SIMET: an insider's perspective. IDA Document D-1661. Institute for Defense Analyses, Washington, D.C
40. Miller DC (2015) SIMNET and beyond: a history of the development of distributed simulation. In: Proceedings of the interservice/industry training, simulation, and education conference. Orlando, FL
41. Pope AR (1991) The SIMNET network and protocols. Report No. 7627, BBN Systems and Technologies, Cambridge, MA
42. Thorpe J (2010) Trends in modeling, simulation and gaming: personal observations about the past thirty years and speculation about the next ten. In: Proceedings of the interservice/industry training, simulation, and education conference. Orlando, FL
43. DIS Steering Committee (1994) The DIS vision, a map to the future of distributed simulation. Institute for Simulation and Training
44. Hofer RC, Loper ML (1995) DIS today [Distributed interactive simulation]. Proc IEEE 83(8): 1124–1137
45. Loper M, Goldiez B, Smith S (1993) The 1992 I/ITSEC distributed interactive simulation interoperability demonstration. In: Proceedings of the interservice/industry training, simulation, and education conference. Orlando, FL
46. Shiflett JE (2013) Observations on the development and implementation of distributed simulation. In: Proceedings of the interservice/industry training, simulation, and education conference. Orlando, FL
47. Voss LD (1993) A revolution in simulation: distributed interaction in the '90s and beyond. Pasha Publications Inc., Virginia
48. Page EH, Canova BS, Tufarolo JA (1997) A case study of verification, validation and accreditation for advanced distributed simulation. ACM Trans Model Comput Simul 7(3)
49. Weatherly R, Seidel D, Weissman J (1991) Aggregate level simulation protocol. In: Proceedings of the summer computer simulation conference. Baltimore, MD
50. Kuhl F, Weatherly R, Dahmann J (1999) Creating computer simulation systems: an introduction to the high level architecture. Prentice Hall

51. Powell ET, Noseworthy JR (2014) The test and training enabling architecture (TENA). Engineering principles of combat modeling and distributed simulation. Wiley
52. Lin KC (1995) Dead reckoning and distributed interactive simulation. In: Distributed Interactive simulation systems for simulation and training in the aerospace environment: a critical review, vol 10280. International Society for Optics and Photonics, p 1028004
53. Fujimoto RM (2000) Parallel and distributed simulation systems. Wiley, New York
54. Jones RM, Laird JE, Nielsen PE (1996) Moving intelligent automated forces into theater-level scenarios. In: Proceedings of the sixth conference on computer generated forces and behavioral representation
55. McGregor D, Brutzman D (2008) Open-DIS: an open source implementation of the DIS protocol for C++ and Java. In: proceedings of the fall simulation interoperability workshop. Orlando, FL
56. Gritton J, Kotick D, Fraas G, Dean C (2016) Washington, we have a problem: the foundation for live virtual constructive (LVC) exercises requires fixing!. In: Proceedings of the interservice/industry training, simulation and education conference
57. Fujimoto RM (1998) Time management in the high level architecture. SIMULATION 71(6): 388–400
58. Hollenbach JW (2009) Inconsistency, neglect and confusion: a historical review of DoD distributed simulation architecture policies. In: Proceedings of the spring simulation interoperability workshop. Orlando, FL
59. Möller B, Dubois A, Le Leydour P, Verhage R (2014) RPR FOM 2.0: a federation object model for defense simulations. In: Proceedings of the fall simulation interoperability workshop. Orlando, FL
60. Henninger A, Cutts D, Loper M, Lutz R, Richbourg R, Saunders R, Swenson S (2008) Live virtual constructive architecture roadmap (LVCAR) final report. Modeling and simulation coordination office project no. 06OC-TR-001
61. Loper M, Cutts D (2008) Live virtual constructive (LVC) architecture roadmap (AR) comparative analysis of standards management. DoD Office of Security Review, Washington, DC
62. Gomes C, Oakes BJ, Moradi M, Gamiz AT, Mendo JC, Dutre S, Denil J, Vangheluwe H (2019) HintCO—hint-based configuration of co-simulations. In: Obaidat M, Ören T, Szczerbicka H (eds) Proceedings of the international conference on simulation and modeling methodologies, technologies and applications, Springer.
63. Calvin J, Dickens A, Gaines B, Metzger P, Miller D, Owen D (1993) The SIMNET virtual world architecture. In: Proceedings of IEEE virtual reality annual international symposium, IEEE, pp 450–455
64. Miller DM, Thorpe JA (1995) SIMNET: the advent of simulator networking. Proc IEEE 83(8):1114–1123