DISTRIBUTED SIMULATION SCIENCE

Margaret L. Loper
Georgia Tech Research Institute
margaret.loper@gtri.gatech.edu

ABSTRACT
Computing technology has advanced dramatically over the last twenty years, enabling new applications for networked simulation. Along with these applications are architectures and standards that support the interoperability of heterogeneous simulations. This paper begins by looking at the historical roots of distributed simulation, from an architecture and policy perspective. The paper then examines distributed simulation as a technology, which is based on the science of distributed systems. The paper identifies two key characteristics of distributed systems and then describes the modern distributed simulation architectures based on these characteristics.

Keywords: distributed simulation, interoperability standards, distributed systems

1. INTRODUCTION

Early computer simulations were limited from several perspectives. Most simulators required identical computer hardware and software in order to operate and create the environment. In this sense they were closed systems, and not expandable. Also computer networks were synchronous; the fixed nature of synchronous systems using shared memory and time slicing of data arrivals made adding simulators to these networks impractical. Technically it was a challenge to separate simulator capabilities so that functions could be performed autonomously.

This changed with the evolution of computer and communications technologies. The late 1960's and early 1970's brought the development of several key technologies that enabled distributed simulation. The UNIX operating system brought the ability to handle asynchronous events, such as non-blocking input/output and inter-process communication. That was followed by expanded computer-to-computer communications with the creation of the Advanced Research Projects Agency Network (ARPANET). The ARPANET enabled a computer to communicate with more than one machine by transmitting packets of data (datagrams) on a network link. The invention of Ethernet technology enabled the connectivity of a variety of computer systems within a local geographical area. Thus simulators no longer needed to be located in the same room, but could be distributed. Then, in 1974, the Transmission Control Protocol and the Internet Protocol (TCP/IP) emerged as a means to enabled intercomputer communications. This development supported heterogeneous computer communications, which would change simulators from being bounded by their computational resources to being bounded by their ability to send and receive information.

The remainder of the paper examines distributed simulation as a technology. It begins by looking at the historical roots of distributed simulation, from an architecture and policy perspective. It then reviews the science of distributed systems and identifies two key characteristics that apply to distributed simulation. Using these characteristics, the paper describes the modern distributed simulation architectures in use today.

2. DISTRIBUTED SIMULATION BEGINS

As computing technology advanced, applications for networked simulation emerged. These applications included inter-crew training, live tests and training, and analysis. For inter-crew training, improving tactical proficiency and training large groups was a growing need. In live tests and training, reducing the number of events was desired due to safety, environmental and cost concerns. In the discipline of analysis, there was a need for individual subsystem evaluation and fine-tuning of tactics and strategies. The emerging application areas and the advances in computing and communication technology led to the first generation of distributed simulation architectures.

Initiated in 1983 by the Defense Advanced Research Projects Agency (DARPA), the SIMulator NETworking (SIMNET) project was the first attempt to exploit the developments in communications technology for simulation (Miller and Thorpe 1995). SIMNET emphasized tactical team performance in a battlefield context, including armor, mechanized infantry, helicopters, artillery, communications, and logistics. Combatants could visually see each other "out the window" and communicate with each other over radio channels. This distributed battlefield was based on selective fidelity (only provide features for inter-crew skills training), asynchronous message passing, commercial computer networks, and replicated state information. SIMNET was based on six design principles:

- *Object/Event Architecture*; the world is modeled as a collection of objects which interact using events.

ISBN 978-88-903724-3-8

- *Common Environment*; the world shares a common understanding of terrain and other cultural features.
- *Autonomous Simulation Nodes*; simulations send events to other simulations and receivers determine if that information is relevant.
- *Transmission of Ground Truth Information*; each simulation is responsible for local perception and modeling the effects of events on its objects.
- *Transmission of State Change Information*; simulations transmit only changes in the behavior of the object(s) they represent.
- *Dead Reckoning Algorithms*; simulations extrapolate the current position of moving objects based on its last reported position.

The first platoon-level system, as shown in Figure 1, was installed in April 1986 and over 250 networked simulators at eleven sites were transitioned to the U.S. Army in 1990. Two mobile platoon sets (in semi-trailers) were delivered to the Army National Guard in 1991.

Figure 1: SIMNET Simulators at Fort Knox, KY

Among the many contributions of the SIMNET program was the invention of semi-automated forces (SAF). The purpose of a SAF simulation was to mimic the behavior of different objects on the battlefield, whether vehicles or soldiers. Realizing it was impractical to have large numbers of operators to control both friendly and opposing forces, COL James Shiflett created the idea of SAF as a way to put opposing forces on the battlefield. His inspiration for SAF was “Night of the Living Dead”, a movie in which teenagers are attacked by zombies. COL Shiflett wanted a simulation that could produce a large number of “dumb” targets (i.e., zombies) to roam the battlefield and provide targets for SIMNET operators (Loper 2008). Eventually SAF simulations turned into something much more intelligent; they could plan routes, avoid obstacles, stay in proper formations, and detect and engage targets.

Soon after the SIMNET project, DARPA recognized the need to connect aggregate-level combat simulations. The Aggregate Level Simulation Protocol (ALSP) extended the benefits of distributed simulation to the force-level training community so that different aggregate-level simulations could cooperate to provide theater-level experiences for battle-staff training. In contrast to the SIMNET simulators, these simulations were event-stepped and maintaining causality was a primary concern. The ALSP program recognized that various time management schemes and more complex simulated object attribute management requirements were needed. The requirements for ALSP were derived from the SIMNET philosophy (Weatherly, Seidel, and Weissman 1991) and included:

- Simulations need to be able to *cooperate over a common network* to form confederations.
- Within a confederation, *temporal causality* must be maintained.
- Simulations should be able to *join and exit a confederation* without major impact on the balance of the participating simulations.
- The system should be network-based with *no central controllers* or arbitrators.
- *Interactions do not require knowledge of confederation participants* and should support an object-oriented view of interactions.

3. LIFE AFTER SIMNET – THE NEED FOR STANDARDS

Several efforts to evaluate simulation technology during this timeframe supported and encouraged the need to develop and invest in distributed simulation. The Defense Science Board (DSB) task force on *Computer Applications to Training & Wargaming* stated “Computer-based, simulated scenarios offer the only practical and affordable means to improve the training of joint operational commanders, their staffs, and the commanders and staffs who report to them.” (DSB 1988) This was followed by the report on *Improving Test and Evaluation Effectiveness*, which found that Modeling & Simulation (M&S) could be an effective tool in the acquisition process throughout the systems life cycle, especially if employed at the inception of the system's existence. (DSB 1989)

Then in 1991, the potential for distributed simulation for the military was realized in an operational context. The Battle of 73 Easting was a tank battle fought during the Gulf War between the U.S. Army and the Iraqi Republican Guard (Krause 1991). Despite being alone, outnumbered and out-gunned, the 2nd Armored Cavalry (ACR) struck a decisive blow destroying Iraqi tanks, personnel carriers and wheeled vehicles during the battle. The 2nd ACR had trained intensely before the battle both in the field and on SIMNET. Immediately, SIMNET’s potential for network training was confirmed.

The following year, the DSB looked at the impact of advanced distributed simulation on readiness,

ISBN 978-88-903724-3-8

training and prototyping (DSB 1993). They concluded that distributed simulation technology could provide the means to substantially improve training and readiness; create an environment for operational and technical innovation for revolutionary improvements; and transform the acquisition process.

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 soon 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 quickly evolved to include a broader range of technology areas. In 1996 the DIS Workshops transformed itself into a more functional organization called the Simulation Interoperability Standards Organization (SISO). An international organization, SISO is dedicated to the promotion of M&S interoperability and reuse for the benefit of a broad range of M&S communities.

One of the lasting contributions introduced during the time of the DIS Workshops was the definition of Live, Virtual, and Constructive (LVC) simulations (the term LVC was originally coined by GEN Paul Gorman). Live simulation refers to M&S involving real people operating real systems (e.g., a pilot flying a jet). 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). The LVC taxonomy is a commonly used way of classifying models and simulation.

4. DISTRIBUTED SIMULATION SCIENCE

Distributed simulation technology is based on the science of distributed systems. A distributed system is a collection of independent computers that appear to the users of the system as a single computer (Tanenbaum 1995). This definition has two aspects. The first one deals with hardware: the machines are autonomous; the second deals with software: the users think of the system as a single computer. This characterization provides a good foundation for distributed simulation technology. The goal of a distributed simulation is to create the illusion in the minds of the users that the entire network of simulations is a single system rather than a collection of distinct machines. Therefore, understanding how to separate the hardware and software design issues is key to developing the technology.

There are numerous challenges associated with building software to support distributed simulation. These include transparency, openness, scalability, performance, fault tolerance and security. Transparency is specifically important as it refers to hiding the distribution of components, so the system is perceived as "whole" and not a collection of "independent" simulations. Tools are needed to support the construction of distributed simulation software, specifically protocols that support the patterns of communication as well as naming and locating simulation processes.

There are two types of characteristics that distinguish the basic patterns of communication in distributed simulations: communication mechanisms and event synchronization. Communication mechanisms refer to the approach for exchanging data among two or more simulations. This includes message passing, shared memory, remote procedure call and remote method invocation. With message passing, there are several variations of delivery depending on the number of receivers. Data can be sent unicast to individual simulations, broadcast to every simulation, or multicast to a selected subset of simulations. Mechanisms such as publish/subscribe can also be used to define subsets of potential receivers.

Event synchronization refers to the approach for synchronizing the sending and receiving of data among the participants of a distributed simulation. Important properties include time, event ordering and time synchronization. Each simulation in a distributed simulation is assumed to maintain an understanding of time. That can include an informal relationship or a very strict adherence to a simulation or wall clock. In either case, simulations assign a timestamp to each message it generates. Event ordering refers to the way in which events are delivered to each simulation. There are several choices. Receive order delivers events regardless of the message time stamp and its relationship to the global distributed system. Timestamp order delivers events in an order directly related to a global interpretation of time. Time synchronization is related to both time and event ordering in that it's concerned with the global understanding of time in the distributed system. If global time is needed, there are a number of conservative and optimistic synchronization algorithms that can be used to achieve this state.

Communication mechanisms and event synchronization can be implemented in one of two ways: by individual simulations or by an operating system. There are three types of operating system commonly used in distributed systems. A network operating system is focused on providing local services to remote clients, and a distributed operating system focuses on providing transparency to users. Middleware combines the scalability and openness of a network operating system and the transparency and ease of use of a distributed operating system to provide general-purpose services. There are a number of trade-offs with the different approaches, including

ISBN 978-88-903724-3-8

performance, scalability and openness. Modern distributed simulation has implemented a range of these approaches.

5. LIVE VIRTUAL CONSTRUCTIVE SIMULATION ARCHITECTURES

The most widely used LVC simulation architectures in the DoD are Distributed Interactive Simulation (DIS), High Level Architecture (HLA) and Test and Training Enabling Architecture (TENA). A fourth architecture exists but will not be covered in this paper. The Common Training and Instrumentation Architecture (CTIA) was developed to link a large number of live assets requiring a relatively narrowly bounded set of data for purposes of providing After Action Reviews on Army training ranges in the support of large-scale exercises. A description of CTIA can be found in (Henninger, et al 2008).

This second generation of distributed simulation architectures has evolved over the last 20 years using different technologies, standards and funding strategies. The following sections give a brief description of the architectures, characterizing its approach for communication and event synchronization.

5.1. Distributed Interactive Simulation

"The primary mission of DIS is to define an infrastructure for linking simulations of various types at multiple locations to create realistic, complex, virtual "worlds" for the simulation of highly interactive activities." (DIS 1994) Distributed Interactive Simulatio is based on the fundamental design principles of SIMNET. The goal of DIS is to create a common, consistent simulated world where different types of simulators can interact. Central to achieving this goal are protocol data units (PDUs); standard messages exchanged to convey state about entities and events. The PDUs comprise object data related to a common function, for example entity state, fire, detonation, and emissions were all frequently used PDUs. The Institute of Electrical and Electronics Engineers (IEEE) approved the first DIS standard in 1993 with 10 PDUs; the most recently published standard has 67 PDUs (IEEE 1278.1a 1998).

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 (McGregor and Brutzman 2008). The first DIS demonstration was held at the 1992 Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) in San Antonio, TX. The demo included 20 companies, 25 simulators, and one long haul connection. The network layout for the demonstration is shown in Figure 2. A minimal set of PDUs (Entity State, Fire and Detonation) were used, and the interaction among participants was focused mainly on unscripted free-play (Loper, Goldiez, and Smith 1993).

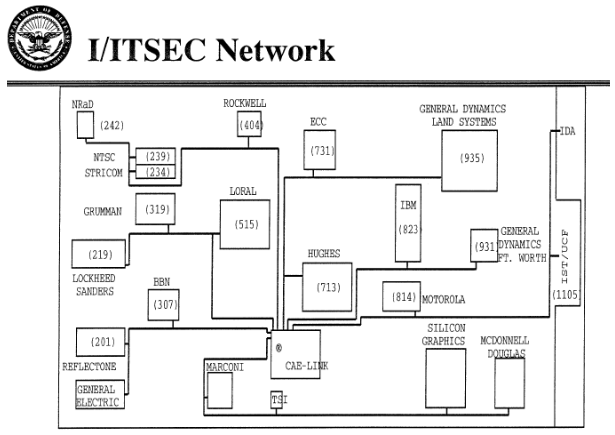

Figure 2: 1992 I/ITSEC DIS Demo Network

From a distributed system viewpoint, DIS is based on the idea that the network and simulators are integrated, i.e. there is minimal transparency. All communication about simulation entities and their interactions occurs via the PDUs. Reasonably reliable delivery is sufficient; dead reckoning algorithms are robust, so 1-2% missing datagram (randomly distributed) does not have an adverse impact on performance. As a result, most PDUs are sent using the best-effort user datagram protocol (UDP). The network is assumed to provide a certain level of assured services including, 300 msec end-to-end latency for "loosely coupled" interactions and 100 msec total latency for "tightly coupled" interactions (IEEE 1278.2 1995). Due to the potential for high latency in wide area networks, DIS is best for exercises on local area networks.

Interaction among DIS simulations is peer-to-peer and occurs using a message-passing paradigm. Since PDUs are broadcast to everyone on the network, bandwidth and computing resources can be consumed processing data that is not relevant to a specific simulation. A study of multicast communications occurred in the early 90's, with the idea of developing a new protocol for highly interactive applications. Developing a new protocol proved problematic and was abandoned. However, progress was made in understanding how to create multicast groups. One of the most commonly understood approaches to grouping information was called *Area of Interest* (Macedonia, et al 1995). Multicast was difficult to implement in DIS due to the lack of middleware or a distributed operating system, which could provide transparency to the simulations.

Time in DIS simulations is managed locally. Each simulation advances time at its own pace and clocks are managed locally using a local understanding of time. There is no attempt to manage time globally. Each PDU has a timestamp assigned by the sending simulation and PDUs are delivered to simulations in the order received. Simulations provide ordering locally, based on their understanding of time.

ISBN 978-88-903724-3-8

5.2. High Level Architecture

The High Level Architecture (HLA) program emerged in the mid-90s based on several assumptions. The first premise is no one simulation can solve all the DoD functional needs for modeling and simulation. The needs of the users are too diverse. Changing user needs define the second premise; it is not possible to anticipate how simulations will be used in the future or in which combinations. It is important, therefore, to think in terms of multiple simulations that can be reused in a variety of ways. This means as simulations are developed, they must be constructed so that they can be easily brought together with other simulations, to support new and different applications.

These assumptions have affected the HLA design in several ways. Clearly, the architecture itself must have modular components with well-defined functionality and interfaces. Further, the HLA separated the functionality needed for individual simulations (or federates) from the hardware infrastructure required to support interoperability. The HLA architecture is defined by three components:

- *Rules* that simulations must obey to be compliant to the standard
- *Object Model Template (OMT)* specifies what information is communicated between simulations and how it is documented
- *Interface Specification* document defines a set of services that simulators use to communicate information

The HLA standards began in 1995 under a government standards process managed by the Architecture Management Group. The DoD adopted the baseline HLA architecture in 1996 and the standards were moved to an open standards process managed by SISO (IEEE 1516 2010; IEEE 1516.1 2010; IEEE 1516.2 2010).

From a distributed system viewpoint, HLA is based on 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. Data can be sent using both best effort (UDP) and reliable (TCP) internetwork protocols. An important distinction is that the HLA is not the same as the RTI. The RTI is an implementation of the HLA Interface standard, and thus there can be many different RTIs that meets HLA Interface standard. From an implementation perspective, HLA follows a commercial business model. There have been a variety of open-source initiatives, but none have produced an HLA compliant RTI.

In contrast to the static DIS PDUs, HLA uses the concept of OMTs to specify the information communicated between simulations. This enables simulation users to customize the types of information communicated among simulations based on the needs of the federation (what DIS called an exercise). When the OMT is used to define the data for a federation, the Federation Object Model (FOM) describes shared information (e.g., objects, interactions) and inter-federate issues (e.g., data encoding schemes). It didn't take long, however, for the community to understand the difficulty in developing FOMs. This led to the emergence of reference FOMs (SISO 2001), a mechanism for representing commonly used information, and Base Object Models (BOMs), a mechanism for representing a single set of object model data (SISO 1998).

From a communications perspective, HLA learned that broadcasting information to all simulations has serious implications on performance. The HLA defined a publication/subscription paradigm, whereby producers of information describe data it can produce and receivers describe data it is interested in receiving. The RTI then matches what is published to what has been subscribed. This approach maximizes network performance by allowing individual simulations to filter data it wants to receive at many different levels.

The HLA does include time management services to support event ordering. 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. Global time advance and event ordering is implemented by means of synchronization algorithms. The HLA interface specification supports the two commonly defined approaches: conservative and optimistic. While HLA provides global time management, use of these services is not required. Simulations can chose to advance time at its own pace, not synchronized with other simulations.

5.3. Test & Training Enabling Architecture

The Test and Training Enabling Architecture (TENA) emerged in the late 90's, after the HLA initiative was underway. The purpose of TENA is to provide the architecture and the software implementation necessary to do three things. First, TENA enables interoperability among Range systems, Facilities, Simulations, and C4ISR (Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance) systems in a quick, cost-efficient manner. It also fosters reuse for Range asset utilization and for future developments. Lastly, TENA provides composability to rapidly assemble, initialize, test, and execute a system from a pool of reusable, interoperable elements.

The principles of the TENA architecture include constrained composition, dynamic run-time characterization, subscription service, controlled information access, and negotiated quality of service. Constrained composition refers to the ability to compose the system for specific intended purposes that may be either transitory or permanent in nature.

ISBN 978-88-903724-3-8

Constraints apply to use of assets including physical proximity and location, coverage regions, performance capabilities, and subsystem compatibility. Dynamic, run-time characterization is focused on responding to many allowable compositions and permitting rapid reconfigurations. This is accomplished by establishing methods for self-description of data representations prior to or concurrent with data transfer or negotiating representation issues before operation starts. Similar to HLA, the subscription service is an object-based approach to data access, which matches producers and consumers of information. Due to the nature of many range assets, controlled information access is particularly important. Levels of access allow users to limit information access to a desired subset of all users. Since some services have significant performance and cost implications (e.g., data streams with large capacity requirements or strict latency tolerance), users can request specialized assets be allocated when needed. The negotiated quality of service protocols relies on the principal of separation of control information from data.

The TENA project uses a government standards process and is managed by Architecture Management Team (AMT). The AMT controls implementation content and Government members of the AMT recommend implementation changes. As such, no open standards have been published for TENA, however they do follow a formal process for standardizing object data.

From a distributed systems view, TENA separates the functionality of range assets from the infrastructure required to communicate among assets using middleware. The TENA Middleware is a common communication mechanism across all applications, providing a single, universal data exchange solution. Data exchanged among range assets is defined in object models, which can be sent using both best-effort (UDP) and reliable (TCP) internetwork protocols. A logical range object model is defined for a given execution, and can include both standard (time, position, orientation, etc.) and user-defined objects.

The TENA Middleware combines several communication paradigms, including distributed shared memory, anonymous publish-subscribe, remote method invocations, and native support for data streams (audio, video, telemetry, and tactical data links). Central to TENA is the concept of a Stateful Distributed Object (SDO) (Noseworthy 2008). This is a combination of a CORBA (Common Object Request Broker Architecture) distributed object with data or state. It is disseminated using a publish-subscribe paradigm, and subscribers can read the SDO as if it were a local object. An SDO may have remotely invocable methods.

Given the nature of real-time range assets, there is no requirement for time management to support event ordering. Messages are delivered to assets in the order they are received. The clock services defined in TENA are to manage time issues for the test facility. This includes synchronization and time setting services, as well as maintaining a global clock for exercises.

6. SIMULATION INTEROPERABILITY

Modeling and simulation interoperability is defined as "the ability of a model or simulation to provide services to and accept services from other models and simulations, and to use the services so exchanged to enable them to operate effectively together". Interoperability exists when different systems exhibit the "same" behavior when stimulated by a set of standard procedures. (DoD 2010). One commonly accepted approach for describing interoperability is the Levels of Conceptual Interoperability Model (LCIM). As shown in Figure 3, the LCIM identifies seven levels of interoperability among participating systems and the complexity of interoperations (Tolk 2003). The LCIM associates the lower layers with the problems of simulation interoperation while the upper layers relate to the problems of reuse and composition of models.

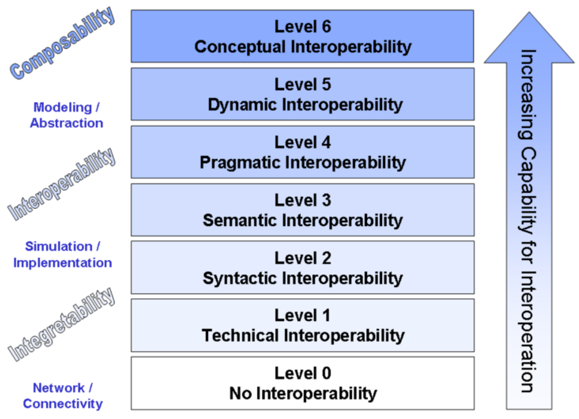
Composability
Modeling / Abstraction
Interoperability
Simulation / Implementation
Integratability
Network / Connectivity
Level 6
Conceptual Interoperability
Level 5
Dynamic Interoperability
Level 4
Pragmatic Interoperability
Level 3
Semantic Interoperability
Level 2
Syntactic Interoperability
Level 1
Technical Interoperability
Level 0
No Interoperability
Increasing Capability for Interoperation

Figure 3: Levels of Conceptual Interoperability Model

DIS, HLA, and TENA are solutions focused on the lower-layers of the LCIM. Since DIS, HLA, and TENA-based federations are not inherently interoperable with each other, additional steps are needed to enable effective communication among those simulations. These steps typically involve using gateways or bridges between the various architectures. While effective, these approaches can introduce increased risk, complexity, cost, level of effort, and preparation time into the simulation event.

Gateways and bridges, however, do not address the issues of reuse and composition associated with the upper layers of the LCIM. As stated in (Tolk 2003), "simulation systems are based on models and their assumptions and constraints. If two simulation systems are combined, these assumptions and constraints must be aligned accordingly to ensure meaningful results". Thus ability to reuse supporting models, personnel (expertise), and applications across the different architectures is limited.

The lack of interoperability between the different architectures introduces a significant and largely unnecessary barrier to the integration of live, virtual, and constructive simulations. This barrier needs to be greatly reduced or eliminated.

ISBN 978-88-903724-3-8

7. CONCLUSIONS

Distributed simulation architectures in use within the DoD today have all been designed to meet the needs of one or more user communities. These architectures continue to evolve and mature based on changing requirements. The existence of multiple architectures allows users to select the methodology that best meets their individual needs. It also provides an incentive for architecture developers and maintainers to competitively keep pace with technology and stay closely engaged with emerging user requirements (Henninger, et al 2008).

One of the challenges in achieving the transparency desired in distributed simulation however is that multiple architectures exist. 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. Integration is typically achieved through gateway solutions, which can often restrict users to a limited set of capabilities that are common across the architectures. The successful integration of distributed simulations will continue to rely upon the development of simulation standards.

Despite the advances in distributed simulation technology and standards, challenges remain. In a 2008 survey on future trends in distributed simulation, the most promising areas of research for the simulation community were identified as distributed simulation middleware, human-computer-interfaces, and the semantic web/interoperability (Strassburger, Schulze, and Fujimoto 2008). Within simulation middleware, the greatest needs identified were plug-and-play capability, standardization and interoperability between different standards, semantic connectivity and ubiquity (accessible anywhere with any device).

The results of this survey combined with the findings of the Live Virtual Constructive Architecture Roadmap panel (Henninger, et al 2008) define the needs for the next generation of distributed simulation. The DoD has been a driving force in shaping the technology and standards for nearly 30 years, and they will continue to have a major role defining the way forward.

REFERENCES

Defense Science Board, 1988. Computer Applications to Training and Wargaming. Final Report, ADA199456, May.

Defense Science Board, 1989. Improving Test and Evaluation Effectiveness. Final Report, ADA274809, December.

Defense Science Board, 1993. Simulation, Readiness and Prototyping. Final Report, ADA266125, January.

DIS Steering Committee, 1994. The DIS Vision: a map to the vision of distributed simulation. IST Technical Report, May.

Henninger, A, Cutts, D, Loper, M, Lutz, R, Richbourg, R, Saunders, R, Swenson, S. Live Virtual Constructive Architecture Roadmap (LVCAR) Final Report. M&S CO Project No. 06OC-TR-001, September 2008.

IEEE 1278.1a – 1998. Standard for Distributed Interactive Simulation - Application Protocols, Version 6 (amendment to IEEE 1278.1-1995).

IEEE 1278.2 – 1995. Standard for Distributed Interactive Simulation - Communication Services and Profiles.

IEEE 1516 – 2010. Standard for Modeling and Simulation High Level Architecture - Framework and Rules.

IEEE 1516.1 – 2010. Standard for Modeling and Simulation High Level Architecture - Federate Interface Specification.

IEEE 1516.2 – 2010. Standard for Modeling and Simulation High Level Architecture - Object Model Template (OMT) Specification.

Krause, M., 1991. The Battle of 73 Easting, 26 February 1991, a historical introduction to a simulation. Draft report, US Army Center of Military History, May 2 May.

Loper, M., 2008. *Interview with COL James Shiflett on History of Distributed Simulation*. Live Virtual Constructive Architecture Roadmap (LVCAR) project. Report, M&S CO Project No. 06OC-TR-001, September.

Loper, M., Goldiez, B., and Smith, S., 1993. The 1992 I/ITSEC Distributed Interactive Simulation Interoperability Demonstration. *Proceedings of the 15th Interservice/Industry Training Systems and Education Conference*. November 29 - December 2, Orlando (Florida USA).

Macedonia, M, Zyda, M, Pratt, D, Brutzman, D, Barnham, P., 1995. Exploiting Reality with Multicast Groups. *IEEE Computer Graphics and Applications*, 15 (5), 38-45.

McGregor, D, Brutzman, D., 2008. Open-DIS: An Open Source Implementation of the DIS Protocol for C++ and Java. *Proceedings of the Fall Simulation Interoperability Workshop*. 15-19 September, Orlando (Florida USA).

Miller, D. and Thorpe, J., 1995. SIMNET: the advent of simulator networking. *Proceedings of the IEEE*, 83 (8), 1114 – 1123.

DoD, 2010. *DoD Modeling and Simulation (M&S) Glossary*. Available from: http://www.msco.mil/ [accessed 15 July 2011].

Noseworthy, J. R., 2008. The Test and Training Enabling Architecture (TENA) Supporting the Decentralized Development of Distributed

ISBN 978-88-903724-3-8

Applications and LVC Simulations. *Proceeding DS-RT '08 Proceedings of the 2008 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications*, pp. 259-268. October 27-29, Vancouver, (British Columbia, Canada).

Simulation Interoperability Standards Organization, 1998. Reference FOM Final Report. SISO-REF-001-1998, March 9.

Simulation Interoperability Standards Organization, 2001. BOM Study Group Final Report. SISO-REF-005-2001, May 15.

Strassburger, S, Schulze, T, Fujimoto, R., 2008. Future Trends In Distributed Simulation and Distributed Virtual Environments: Results Of A Peer Study. *Proceedings of the Winter Simulation Conference*. December 7-10, Miami (Florida, USA).

Tanenbaum, AS., 1995. *Distributed Operating Systems*. 1st Edition New Jersey: Prentice Hall.

AUTHORS BIOGRAPHY

Margaret Loper is the Chief Scientist for the Information & Communications Laboratory at the Georgia Tech Research Institute. She holds a Ph.D. in Computer Science from the Georgia Institute of Technology, a M.S. in Computer Engineering from the University of Central Florida, and a B.S. in Electrical Engineering from Clemson University. Margaret's technical focus is parallel and distributed simulation, and she has published more than 50 papers as book chapters, journal contributions, or in conference proceedings. She is a senior member of the IEEE and ACM, and member of the Society for Modeling and Simulation. She is a founding member of the Simulation Interoperability Standards Organization (SISO) and received service awards for her work with the Distributed Interactive Simulation (DIS) and High Level Architecture (HLA) standards and the DIS/SISO transition. Her research contributions are in the areas of temporal synchronization, simulation testing, and simulation communication protocols.

ISBN 978-88-903724-3-8