Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

A SOCIETY OF SIMULATION APPROACH TO DYNAMIC INTEGRATION OF SIMULATIONS*

Alok Chaturvedi

Purdue Homeland Security Institute
Krannert Graduate School of Management
Purdue University
West Lafayette, IN 47907, U.S.A.

ABSTRACT

A Society of Simulations (SoS) approach defines a semantics-based standard for integrating heterogeneous simulations. Heterogeneity, independent development, and cross-domain modeling are characteristics of behavior-based simulations that are built to address complex, non-functional situations. SoS is a dynamic data driven approach wherein emergent behaviors result from specialized simulations grouped together as individual members in a society. Each member simulation independently operates on its own understanding of reality. Members cooperate with each other to achieve the goals of the society while satisfying their own local goals. When members interact, portions of their models of reality coincide. Such interaction and coordination among diverse autonomous simulations is enabled through the implementation of a shared reality. Comprehensive, life-like situations can be modeled by applying the concept of a SoS.

1 THE NEED FOR INTEGRATION

Simulations are increasingly being used for analysis, training, and courses of action development for complex, real world problems, such as emergency preparedness and response, global corporate strategy development and assessment, and military campaign planning for winning the hearts and minds of the population. These problems require multi-disciplinary thinking, multi-scale temporal and spatial representations of forward and inverse problems, multiple analytical points of view, massive numbers of entities representing multiple sides with diverse interests, and emergent behaviors and interactions both among entities and between entities and the environment. Building such simulations from scratch is expensive (hundreds of millions to billions of dollars) and rarely gets completed on time and within budget (US Department of Defense 1997).

The alternative is integration of existing simulations, components, and models. The models originate in specialized fields and characterize phenomena at diverse levels of detail. Furthermore, the components are typically developed independently and continuously, using different semantics and data structures. Applying traditional requirements-driven approaches (Giogini et al. 2004) to building such comprehensive simulations is inadequate in that the dependencies between the various components are only well-defined during the run-time of the simulation.

A Society of Simulations (SoS) is a dynamic data driven approach to integrating simulations across heterogeneous semantics, data formats, and granularities (temporal, spatial, and otherwise). It allows reuse of independently developed components and is scalable. SoS approach specifically addresses situations where the simulations exhibit dynamic characteristics and have heterogeneous requirements. This paper introduces a semantics-based standard for integration with a SoS. The following section defines the concept of a SoS. Section 3 outlines the process of developing a SoS, and Section 4 provides several recent examples of how a SoS has been applied to address real world problems.

2 THE SOCIETY OF SIMULATIONS CONCEPT

A SoS is analogous to a society of people as both are loosely coupled constructs in which independent individuals contribute toward a single societal identity. A society can be defined as an organized group of individuals who associate for common purposes. Likewise, autonomous simulations in a SoS work together to achieve the common goal of modeling the system. Each simulation in a SoS is an autonomously managed member which cooperates with other members to reach its personal goals. In the process of meeting its personal goals, a member contributes to societal goals. Satisfaction of societal goals emerges as all members progress towards their personal goals.

A SoS can be heterogeneous in terms of the data the members produce and consume as well as in terms of how the members advance their simulation times. Concerning heterogeneous data representations, members can have unequal temporal and spatial granularities, be from various

fields of science, require different data, use different syntax, and apply different meanings to the same phenomenon. Despite their nonconformity, members contribute to the goals of the society and benefit from the society's resources.

A SoS approach consists of an information sharing mechanism, a framework of linking distributed component simulations to shared information, and a process of analyzing and assessing a simulation with respect to the overall goals of the society. These three components are referred to as shared reality, members, and liaisons.

Figure 1 illustrates the components of a SoS. Each member is connected to shared reality through a member-specific liaison. Interactions among members result from aspects of reality that are represented in multiple simulations. These shared aspects make up the shared reality component and are accessible by all members. All interactions between members occur as members act on entities within shared reality.

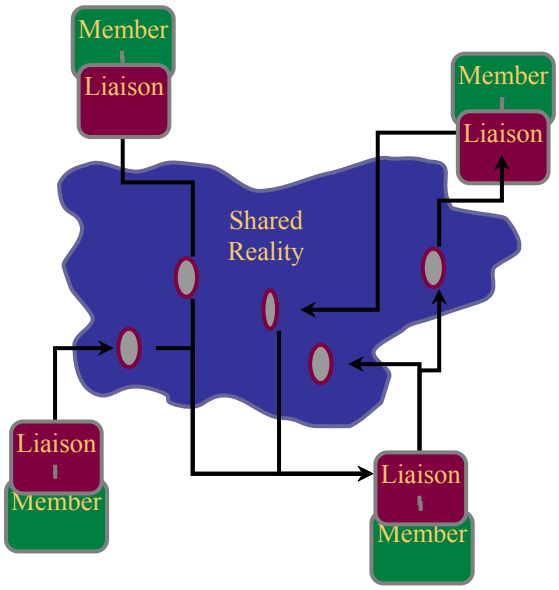

Figure 1: Illustration Showing the Components of a SoS

2.1 Shared Reality

Producer members and consumer members are linked to shared information instead of to each other. Decoupling producers and consumers enables heterogeneous time management styles among members. Consequently, the time management mechanisms employed by producers do not dictate how consumers must manage their times.

Shared reality is distributed, enabling localized exchange of data between dependent members. Each member senses changes to the portions of the shared reality that are relevant to its execution. The regions of the shared reality that a member depends on can change during run time.

What data a member requires and when to incorporate new data during its execution are characteristics of a member and the scenario that is unfolding in the simulation. Therefore, the intelligence for transforming information within shared reality into a form a consumer can digest and for synchronizing a consumer with produced data is pushed from the data exchange mechanism of shared reality onto the member-specific liaisons that connect members to shared reality.

Consequently, shared reality is light weight in the sense that the overhead a member experiences when accessing shared reality is not due to the time management of the other members in the society since shared reality does not manage the members. Nor is the overhead for accessing shared reality proportional to the number of members in the society since shared reality is not keeping track of the members that are accessing its data.

2.2 Members

Each member in a SoS accesses shared reality through a member-specific liaison. A liaison consists of the intelligence needed to interact with and control a member and to interact with the rest of a society. A liaison is configured to use member-specific mechanisms—initializations, inputs, outputs, and control mechanisms. In this way, the same member can be used in different societies and be continuously developed without being forced to address society-specific characteristics, enabling reuse and distributed development.

2.3 Liaisons

All conversions and transformations are performed by the liaisons of the consuming members. Producing members create data within shared reality without accommodating the potentially diverse representations used by other members. By doing so, the overhead of accommodating the various member formats is performed by each consuming member asynchronously. A liaison performs the following tasks:

- Synchronizes the member with data the member depends on.
- Starts, stops, restarts, and checkpoints a member.
- Gathers data from shared reality, transforms its syntax, converts its granularity, and translates its semantics.
- Places the member's outputs into shared reality coupled with semantic information describing the syntax, granularity, and semantics of the data.

A SoS provides a means whereby independent members with diverse representations of information and diverse methods for advancing a member's internal simulation time can interact.

3 ARCHITECTING A SOCIETY OF SIMULATIONS

Building a SoS follows six fundamental steps:

1. *Analyze the society*: The simulation users and the sponsors define and agree on a set of objectives and required simulation members to achieve these objectives.
2. *Analyze the members*: Each member is studied with respect to its potential contributions to the goals of the society, dependencies on the types of information that will be represented in the society, its timing and performance features, and the abstract program interface (API) that a liaison will use to connect to the member. Each input and output of a member is described in terms of three dimensions: semantics, granularity for each dimension, and syntax.
3. *Build shared reality*: A domain-specific ontology is placed within shared reality to facilitate semantic matching that liaisons of consumer members will perform. Producing members extend the ontology as needed to describe the semantics of the data they provide to shared reality. Also, the total capacity for a shared data type can be estimated to determine the required storage space based on the amount of data persistence needed by the members.
4. *Design the liaisons*: A liaison is customized for each member to perform the translation of semantics between the information produced in shared reality the information that satisfies a member's input requirements. A liaison must be equipped to convert granularities when the member consumes data at a different granularity than it is produced. The liaisons are also constructed to perform the task of transforming between syntaxes.
5. *Design the system*: Required resources are allocated for execution of a SoS. The resources include software tools and hardware facilities required for the members to execute, memory for shared reality, initialization data, and required network access for distributed members to access shared reality.
6. *Execute, test and optimize the society*: The society is executed and the output is collected and analyzed to ensure the correctness. Optimistic simulation performance can be tuned to address performance bottlenecks.

Note that a SoS enables distributed development of the member simulations. The goals of the society established in the first step do not dictate how the members are to be designed.

Also note that since the data exchange occurs emergently in a SoS, there is no step to link the members together in an explicit manner. To establish explicit links after global analysis of the entire society would not be scalable as more members are added or as the design of members change.

Describing the ontology of data that will be housed by shared reality enables liaisons for consumer members to perform semantic translation of information. For example, a fire simulation member may consume changes to a building layout, since the layout of items influences the flow of air which impacts the oxygen supplied to a fire. A human simulation member may take actions on doors or windows in an effort to escape. The actions taken by human simulation members must be translated into the movement of certain obstacles in a building layout in order for the human's actions to impact the spread of fire.

3.1 Execution-Time Development of a Society

Since the design of members are decoupled and the semantics of data that resides is shared reality is accessible by all members, new members can join an existing society during execution-time. A new member is analyzed with respect to the society, just as the members gathered at the initialization of a society are analyzed in Step 2 above.

New information produced by the new member is described within the ontology in shared reality, enabling liaisons of consumer members to recognize the relevance of the new data to their members' inputs.

The process of adding new members and having an existing society adjust automatically is not fully automated. Existing liaisons may not have the utilities available to perform the translation, conversion, and transformation necessary to enable the data to be consumed. However, libraries of utilities for many of the common semantic translations, granularity conversions, and syntax transformations developed for one society can often be reused in societies with similar domains.

4 RECENT APPLICATIONS OF A SOCIETY OF SIMULATIONS

The SoS approach has been implemented to address a number of different problems, three of which are described below. The first two instances show how a SoS approach has been applied to integrating a behavior-based simulation modeling human behavior with a HLA federation of tactical, military simulations to achieve the goal of modeling reconstruction operations. The third instance addresses a fire evacuation scenario where the members have hetero-

geneous characteristics and the outcome of the evacuation is an emergent property of the society.

4.1 Multi-National Experiment 4

Synthetic Environment for Analysis and Simulation (SEAS), the behavior-based simulation, generates the Political, Military, Economic, Social, Information and Infrastructure (PMESII) impacts of actions taken at local, national, and global levels. SEAS is described in more detail in Chaturvedi et al. 2004.

Using a SoS, SEAS is integrated with Joint Semi-Automated Forces (JSAF), which models the real-time tactical activity in a specified city or nation. This integration of SEAS and JSAF provides a model of how local events influence populations that in turn can positively or negatively impact stability operations.

Multi-National Experiment 4 used a SEAS-JSAF society to implement a full synthetic world that allows human-in-the-loop (HITL) players to interact with a broad range of actors in a strife torn region of the world, including coalitions of forces, non-government organizations (NGO), local government agencies, terrorist networks, media outlets, internally displaced persons (IDPs), and local administrations. The synthetic world is designed so that the outcomes of players' actions emerge much like they do in the real world.

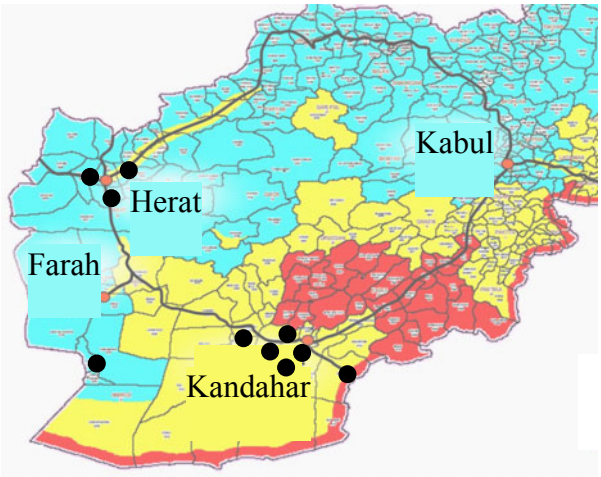

Figure 2: The MNE4 Scenario

The MNE4 scenario investigates the impact that well being of internally displaced persons (IDP) can have on the coalition's objectives. Approximate locations of a number of IDP camps are indicated in Figure 2. The coloring in the map indicate areas of risk with respect to stability and terrorist activity, (blue is low risk, yellow is medium, and red is high). Though the population of the IDP camps is much smaller than the population of the country of interest, health and security issues within IDP camps can require military resources, influence the effectiveness of NGOs, and sway world public opinion.

SEAS is used to construct a virtual international system (VIS) consisting of a number of virtual states. A Virtual State is represented by four primitive constructs: Individuals, Organizations, Institutions, Infrastructures, and Geographies (IOIIG). These four primitives are used to model higher order constructs such as geographical entities (nations, provinces, cities), political systems (type of government, political parties/factions), military (soldiers, institutions), economic system (formal and informal structures), social system (institutions, groups), Information systems (print, broadcast, internet), and critical Infrastructures (banking, oil and gas, electricity, telecommunications, transportation).

For many of the SEAS models used in MNE4, a day tick temporal granularity was used, capturing international alliances, media broadcasts, and changes in the economy, for example. Within the country of interest, a one-to-one-thousand sampling of agents to people was used to represent the population.

However, finer grained models had to be employed to address the health and security issues surrounding IDP camps. Within an IDP camp, an hourly temporal granularity is used to capture health crises and crowd violence that can result from basic needs not being met and ethnic or class differences within a camp. In an IDP camp, each of the approximately 100,000 individuals is modeled as an agent.

The multi-granularity models employed by SEAS were able to interact with real-time events in JSAF through a SoS. The security of an IDP camp is influenced by the presence of coalition forces within reach of the camp. The presence of forces is communicated from JSAF to a JSAF Liaison which, in turn, keeps the representation of troops in shared reality up to date. Additionally, JSAF communicates whenever an explosion occurs, which is also placed within shared reality.

The SEAS-VIS liaison (daily tick granularity) accesses shared reality and takes the explosions into account to determine the security in a region, which impacts the supply of medical and food resources to camps in the area. Changes in the rate of provisions is communicated to shared reality and is consumed by the SEAS-Near-Real-Time (SEAS-NRT, used to model the IDP camps) liaison.

The SEAS-JSAF society used in MNE4 shows the ability of the SoS approach in handling heterogeneous granularities in time, space, and population sampling.

4.2 Urban Resolve 2015 (UR)

A SEAS-JSAF SoS is also being used to facilitate the military decision-making process in complex, urban environments. Initial work on constructing this society was described in Chaturvedi et al. 2005. Many different models

are used by SEAS-VIS in UR than in MNE4 to represent terror cell groups, IED attacks, interconnections among infrastructure, emergent organization restructuring, and militia recruiting, to name a few. Additionally, JSAF and SEAS share more information due to the complex, tactical issues faced in an urban context, such as the locations, health, and activities of key, influential individuals, the damage or repair of selected buildings within the city, illness due to chemical attacks, and tactical activity taken to incite or calm specific people groups in an area.

The UR SoS involves the following members: JSAF, SEAS, a persistent data storage and experiment initialization for multiple runs of the experiment, and a user interface for analyzing and interacting with networks that emerge in real-time. SEAS models the structure of the organizations and leaders' attitudes to the friendly and opposing forces. JSAF represents certain leaders in a detailed city environment. When actions are taken against a leader in JSAF changes to organizations can occur in SEAS. The persistent data storage liaison senses significant changes in shared reality, such as deaths of key individuals or organization restructuring, and persists the change for this experiment run. The user interface also senses changes to leaders and organizations. These changes are immediately propagated to the client visualization to reveal the new organization structure in real time.

4.3 Evacuation Society

A SoS was built (named Evacuation Society) to model the influence of human behavior on an emergency evacuation in the event of a fire, taking into account both the physics of fires as well as the activity and health of the humans evacuating (Foong et al. 2005). The Evacuation Society consists of one fire simulation member and four synthetic human members, all placed within a virtual townhouse. The outcome of an evacuation in the Evacuation Society is emergent, depending on the decisions the human members make as they interact with their environment and with each other.

The fire member is a fluid dynamics simulation modeling the spread of fire across materials and the flow of gases within a layout of obstacles. The human members are built to model a person's implementation of intentions within a physical environment, such as moving around obstacles, rescuing other people, routing to and searching for a location. The human members react to smoke and their health is influenced by toxic fumes from the fire and extreme temperatures. The human members can influence the fire by opening doors and windows, changing the air flow within the house.

The Evacuation Society exhibits heterogeneity in terms of the time management, temporal granularity, and semantics used by a the members. The fire simulation is governed by an optimistic, continuous-time model while the human simulations are conservative, discrete event simulations. The fire member operates at a millisecond of precision whereas the human members operate at a second temporal granularity. The actions taken by the humans on the items in the house are translated by the liaison of the fire member into changes in the layout of the obstacles.

5 RELATED APPROACHES TO INTEGRATION

A variety of approaches have been proposed to integrate diverse simulations. SIMNET is one of the earliest efforts to integrate many autonomous components together in order to build a large, distributed simulation (Miller 1995). SIMNET consists of a bus-like communication network upon which all messages are broadcast to all simulations. An autonomous simulation in SIMNET is not aware of which other simulations are connected to the network. Any simulation that can connect to the network can participate in the simulation. Moreover, simulations can be attached to and removed from the network dynamically during the execution of the simulation. SIMNET does not require a central controller that regulates the time management of the simulations. Rather, each simulation synchronizes itself with every event that is published on the network.

However, the performance of SIMNET is limited by the bandwidth of the network as each simulation in the network consumes events broadcast by all other simulations. The experiences with SIMNET led to the development of the High Level Architecture (HLA) (Fujimoto 2001, IEEE Std 1516-2000, 1516.1-2000, 1516.2-2000, 1516.3-2003). In HLA, time and data management are used to refine the communication a simulation receives based on a publish-subscribe mechanism.

HLA defines an interface between a simulation and a set of management tools that enable interactions among arbitrary simulations. To connect different simulations (called federates in HLA), a Run Time Infrastructure (RTI) is used. The HLA standard specifies the interface between the RTI and constituent federates. A RTI is given only enough knowledge of the specifics of the federates in order to allow them to interact, such as the data a federate subscribes to and what a federate's next time step will be. Federates are not aware that other federates are connected in the same federation. Federates interact only through the RTI. Separating the design of federates from the implementation of the RTI facilitates reuse of federates and the RTI.

To address scalability issues, time and data management mechanisms of the RTI have to be tuned. Computation of the lowest bound time stamp (LBTS) is necessary for enabling parallel execution, but its computation requires coordination among all federates.

Data management can also cause significant overhead in HLA when the portion of data a federate consumes changes dynamically during the simulation. The HLA me-

thod of combining simulations relies on a push-based protocol wherein newly produced data is pushed onto the consumers, resulting in production of duplicate and irrelevant messages. Duplicate messages occur when a federate is notified more than once of a change to data, which occurs when the data manager decomposes the data into too fine of a granularity. Irrelevant messages result from a federate receiving an update that it does not need, which occurs when the data manager decomposes the data into too coarse of a granularity.

Approaches have been proposed to alleviate the time and data management limitations, such as implementing the LBTS computation as a repeated reduction (Fujimoto et al. 2000, Perumalla et al. 2003). For data management, a proposed remedy is to use a hybrid scheme which divides up the data space into a fine grid and then keeps track of the coarser portion of the grid that each federate produces and consumes (Fujimoto et al. 2000). Yet, the RTI is still burdened with the overhead of bookkeeping the local times of the federates and the sets of federates to which updates of data should be broadcast.

The performance of a HLA federation is further limited when the RTI must assume the existence of dependences among the federates at each time step, such as when the federates have no known look ahead values. To synchronize simulations while avoiding delays due to unnecessary blocking, an approach was developed to enable optimistic simulation (Jefferson 1985). Optimistic simulation occurs when a simulation is allowed to progress beyond a simulation time, even if doing so may violate a causality condition. Causality is violated when events are processed out of order.

Time Warp enables optimistic simulation by causing the simulations to roll back to a simulation time in the past when a causality violation occurs. A simulation periodically checkpoints its state and records the messages it sends in each period. When a simulation must roll back to a time in its past, it will restore an old state prior to the roll back time and send out anti-messages (messages that cancel the effects of their counterparts) for every message it had sent after the roll back time. Time Warp avoids much of the centralized time management of HLA by allowing each simulation to manage its own state and time.

The optimistic execution enabled by Time Warp comes at the expense of an overhead in space and time. Memory space is required to checkpoint as the space required to save each simulation's state continues to increase as more time steps are simulated. Execution time overhead is accrued each time a roll back occurs.

To offset the memory use problem, a Global Virtual Time (GVT) is calculated and broadcast to each simulation. The GVT is the minimum time any future roll back can occur, similar to the LBTS calculation of HLA. It can be calculated by taking the minimum simulation time from every simulation and every message that is not yet processed.

Any checkpoint or recorded message before the GVT can be discarded or saved to disk because no roll backs will occur with a simulation time less than the GVT. In order to calculate an accurate GVT, synchronization is required from each simulation, which is essentially a barrier synchronization. The overhead of calculating GVT is partially alleviated using an approximate GVT value through a distributed broadcast.

Besides the overhead required to enable roll backs, Time Warp simulations face the challenge of instability. A simulation becomes unstable when continuous roll backs occur, causing the simulation to fail to complete. In response to the potential instability and to address the overheads involved with frequent checkpointing, Avril and Tropper propose a method of grouping Time Warp simulations into clusters, checkpointing and rolling back at the cluster level (2001). The goal of their techniques is to achieve a balance between memory use and execution time.

Creating a system in which diverse simulations can interact is a complex problem with the many interacting tradeoffs described above. The SIMNET approach is limited by the network bandwidth and the small, packet-like message size allowed. The performance of a HLA-compliant federation can suffer from delays that are not necessary for correct coordination of federates and from unexpected overheads in the RTI mechanisms. The optimistic approach used by Time Warp simulations potentially exhibits instability. Additionally, the Time Warp simulations communicate directly with each other, requiring the producers of messages to know who the consumers are.

A SoS approach specifically addresses simulation integration when the simulations exhibit dynamic characteristics and have heterogeneous requirements. A member in a SoS can be developed independently and concurrently with other members without requiring reprogramming of the other members since the linkages between members are emergent properties of the society instead of presupposed characteristics of the society's design.

6 CONCLUSION

A SoS approach enables solutions to multi-domain problems while facilitating distributed development and independent design of domain-specific simulations. Collaboration among heterogeneous simulations is likened to members of a society working together to achieve common goals. Simulations are regarded as members of a SoS. A SoS enables distributed development since members are autonomously managed. Changes to members do not require changes to the design of a SoS, making distributed development possible. Autonomous management is en-

abled by linking members to information instead of to other members.

Many approaches to simulation integration have reuse, heterogeneity, and scalability as goals. However, the underlying data exchange mechanisms hinder these goals. A SoS approach allows simulations to cooperate yet remain autonomous, an inherent and scalable approach for linking heterogeneous simulations.

In this paper, we have described the concept of a SoS and the basic six step process one can follow to build a SoS. The process outlines how SoS facilitates the integration of heterogeneous and independently developed member simulations.

7 ACKNOWLEDGMENT

This research was partially funded by National Science Foundation DDDAS program grant # CNS-0325846, the Indiana State 21st Century Research and Technology award #1110030618, and several awards from US Joint Forces Command Experimentation Directorate, J9.

REFERENCES

Avril, H., and C. Tropper. 2001. On Rolling Back and Checkpointing in Time Warp. *IEEE Transactions on Parallel and Distributed Systems* 12 (11): 1105—1121.

Chaturvedi, A. R., D. Dolk, R. Chaturvedi, C. Foong, M. Mulpuri, D. Lengacher, et al. 2004. Agent-Based Computational Model of a Virtual International System. In *Proceedings of the Conference on Agent 2004: Social Dynamics: Interaction, Reflexivity and Emergence*. Chicago, IL: 677—692.

Chaturvedi, A., D. R. Snyder, C. M. Foong, and B. Armstrong. 2005. Bridging Kinetic and Non-Kinetic Interactions Over Time and Space Continua. *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2005*. Paper 2123. Orlando, FL: 1—11.

Foong, C., B. Armstrong, D. Dilley, J. Grahn, K. Krull, A. Chaturvedi, et al. 2005. Towards Enabling A Distributed And Scalable Society Of Simulations. *In Proceedings of the 2005 Agent-Directed Simulation Symposium*. Simulation Councils, Inc.: 11—18.

Fujimoto, R., T. McLean, K. Perumalla, and I. Tacic. 2000. Design of High Performance RTI Software. In *Proceedings of the Fourth IEEE International Workshop on Distributed Simulation and Real-Time Applications*. IEEE Computer Society: 89—96.

Fujimoto, R. M. 2001. Parallel simulation: parallel and distributed simulation systems. In *Proceedings of the 33nd Conference on Winter Simulation*. IEEE Computer Society: 147—157.

Giorgini, P., M. Kolp, J. Mylopoulos, and M. Pistore. 2004. The Tropos Methodology: An Overview. *Methodologies And Software Engineering For Agent Systems*. Kluwer Academic Publishing: 1—20.

IEEE Std 1516-2000. 2000. *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—Framework and Rules*: i—22.

IEEE Std 1516.1-2000. 2001. *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—Federate Interface Specification*: i—467.

IEEE Std 1516.2-2000. 2001. *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—Object Model Template (OMT) Specification*: i—30.

IEEE Std. 1516.3-2003. 2003. *IEEE Recommended Practice for High Level Architecture (HLA) Federation Development and Execution Process (FEDEP)*: 1—32.

Jefferson, D. R. 1985. Virtual time. *ACM Transactions on Programming Languages and Systems*. 7 (3): 404—425.

Miller, D. C., and J. A. Thorpe. 1995. SIMNET: The Advent of Simulator Networking. In *Proceedings of the IEEE* 83 (8): 1114—1123.

Perumalla, K. S., A. Park, R. M. Fujimoto, and G. F. Riley. 2003. Scalable RTI-Based Parallel Simulation of Networks. In *Proceedings of the Seventeenth Workshop on Parallel and Distributed Simulation*. IEEE Computer Society: 97—104.

U.S. Department of Defense, Office of the Inspector General. 1997. Requirements Planning for Development, Test, Evaluation, and Impact on Readiness of Training Simulators and Devices, a draft proposed audit report, Project No. 5AB-0070.00, January 10, Appendix D.

AUTHOR BIOGRAPHY

ALOK R. CHATURVEDI is a Professor of Management Information Systems at the Krannert Graduate School of Management, Purdue University; the Director of Purdue Homeland Security Institute, and the founder and Chief Executive Officer of Simulex, Inc. Dr. Chaturvedi has also served as an Adjunct Research Staff Member at the Institute for Defense Analyses (IDA). He received his Ph.D. in Management Information Systems and Computer Science from the University of Wisconsin-Milwaukee. Dr. Chaturvedi has been working on multi-agent synthetic environments for over sixteen years.