Proceedings of the 1999 Winter Simulation Conference
P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, eds.

THE HIGH LEVEL ARCHITECTURE: IS THERE A BETTER WAY?

Wayne J. Davis

General Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801, U.S.A.

Gerald L. Moeller

U.S. Army Material Systems Analysis Activity
Rock Island Arsenal
Rock Island, IL 61299, U.S.A

ABSTRACT

This paper first discusses the basic design approach adopted for the High Level Architecture and the design goals that it addresses in the military simulation arena. Next, the limitations of this architecture are discussed with particular focus on the real-time information requirements needed to support its operation. Finally, the paper discusses HLA's inability to model complex systems with hierarchical command and control structures and the inherent limitations that this deficiency will impose upon the application of futuristic simulation technologies to military applications.

1 INTRODUCTION

Like all planning entities (whether they be individuals or complex organizations), the Department of Defense (DoD) has had to decide whether it is better to repair the old or to begin anew. The design and adoption of the High Level Architecture (HLA) reflects such a decision. Unfortunately, such decisions seldom offer a conclusive best option. Repairing the old may be cheaper in the short run. However, retaining old designs can also impede the adoption of new technologies. In this paper we will explore the situation that has evolved in the adoption of the HLA. We will discuss how the HLA attempts to save and integrate existing military simulation models. We will also discuss the probable limitations that HLA will likely impose as the DoD moves towards futuristic simulation capabilities.

Over the last three decades, the Department of Defense (DoD) has invested significant research and development funds to generate simulation models for the various operational components of the military. This continued development has certainly been influenced by the advancements in simulation and computer technology. That is, more recent models have employed new programming languages and approaches, such as object-oriented programming. Earlier models employed the structured programming approaches that were perceived to be the state-of-the-art at the time. In addition, most of these modeling efforts addressed a singular component of a military mission. Faced with a set of focused simulation models employing various programming technologies, the DoD initiated an effort that would permit these individual simulation models to interact with each other. This was no small feat, particularly because most of these models were not designed to interact with other models and that they often employed different programming styles and languages.

The most recent effort to salvage and integrate the DoD's simulation models was the development of the HLA (see US Department of Defense [1997]). In developing HLA, several design goals were established. Ironically, these goals can be divided into two sets, one set addressing the retention of the existing capabilities, the other looking to the future. The set of goals that addresses the need to salvage existing capabilities includes the following:

- To maximize the reusability of the existing simulations models,
- To permit individual simulation models to be integrated in order to model more complex systems, and
- To allow the individual simulation models to interact in a manner that supports distributed simulation technologies.

The set of design goals that address future capabilities could include:

- To provide an increased capability for modeling command and control (C^2) structures and to assess the effects that a given C^2 structure has upon system performance,
- To provide a flexible framework that will readily permit new technologies to be incorporated in the simulation models, and
- To provide simulation models that can be immediately incorporated into evolving planning and control technologies.

Although HLA does succeed in meeting the goals for salvaging existing simulation capabilities, it is not obvious that HLA is the best solution for addressing future needs. In fact, HLA may actually hinder the adoption of future technologies. The DoD has mandated that all future military models be HLA-compliant. If this mandate is followed, what are the consequences for the military? Is the DoD painting itself into a corner?

Furthermore, the HLA has recently been adopted as the standard for distributed simulation by the Object Modeling Group, Inc. and will likely be adopted by IEEE in the near future. Thus, the HLA is being propagated to simulation community at large. Are we really ready to embrace the HLA as "the standard" for distributed simulations? Have we explored all the alternatives? What are the consequences of this action? These questions must be addressed.

In this paper, we begin by providing a brief overview of the HLA approach. We then discuss some of the immediate issues that arise in the employment of HLA. Finally, we look at the future needs for military simulations and discuss the potential constraints that HLA imposes upon adopting new simulation, planning and control technologies.

2 A BRIEF DESCRIPTION OF THE HLA

Like most simulations, military simulations typically view the target system as a collection of elements and then attempt to model the physical interactions that occur among the elements. (See Figure 1.)

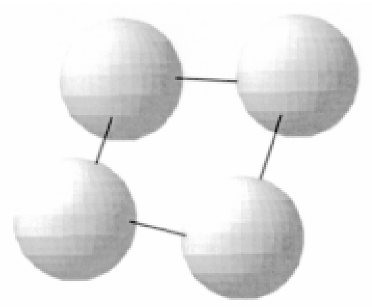

Figure 1: A Simple Schematic of a Simulation Model

The resulting models are designed to consider the internal dynamics among the elements, and do not address elements that are outside the simulation model. For example, a given model might address a mobile artillery group, and not be immediately capable of interacting with another model for an infantry group. Furthermore, the two models may be written in different languages and may employ different programming architectures. How does one integrate these distinct models when s/he attempts to model more complex battle scenarios?

HLA addresses this desire to integrate models by employing an *object model template* (OMT) to encapsulate (or wrap) a given model into a single virtual object called a *federate* (see Figure 2).

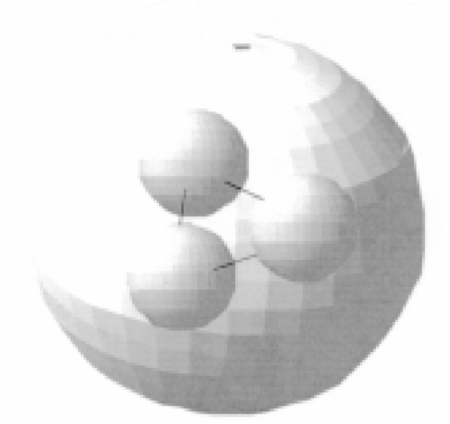

Figure 2: A Schematic Illustration of the Federate (a Virtual Object)

Now assume that we have a collection of simulation models as pictured in Figure 3. Through the use of the OMT, the HLA approach then wraps each of the individual models to form a collection of virtual objects (or federates) as shown in Figure 4. The resulting collection of federates is called a *federation*.

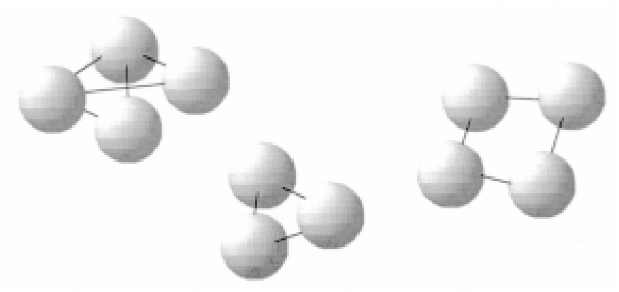

Figure 3: A Schematic Illustrating a Collection of Three Distinct Models

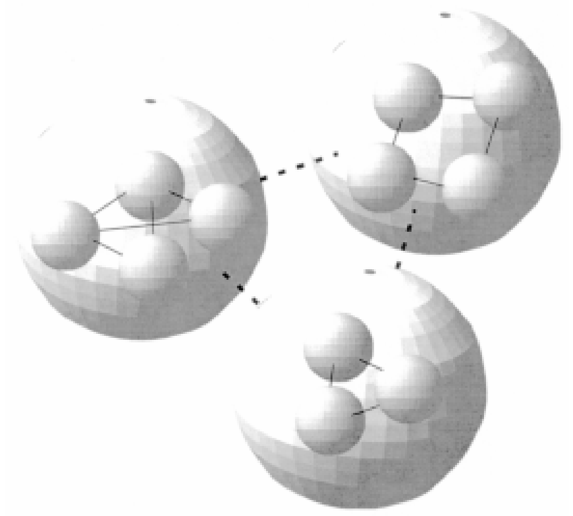

Figure 4: A Schematic for a Federation of Three Distinct Models

Using the HLA approach, all of the elements within each individual model are completely encapsulated within the federate and are hidden from any other federate within the federation. That is, the language and architecture that was employed to program the model within the federate becomes irrelevant to the outside world. However, the internal modeling elements within a given federate are capable of providing their current state information to their wrapper, which then provides the interface of the federate with the outside world.

The next component of the HLA is the *Real-Time Interface* (RTI). The RTI permits one federate to interact with another federate within the federation in real time. (Note, however, that federates within one federation cannot generally interact with federates in another federation.) As is indicated by the dashed lines in Figure 4, the real-time state information associated with the sub-elements within a given federate can be broadcasted to other federates within the federation. Given that the wrapper will be receiving state information from the other federates within the federation, the wrapper that interfaces the federate with the RTI must also be able to transmit external state information to the individual modeling elements that are contained within the federate. Thus, the OMT wrapper that encapsulates the federate provides two-way communication with the modeling elements contained within the federate while the RTI provides for two-way communication among the federates.

A problem arises here. The wrapper for any federate must collect any piece of real-time state information from any element contained within the federate that may be needed by another federate. In this way, the wrapper provides a single, real-time state description for the federate, which is subsequently broadcasted to the other federates in real-time. Figure 4 is deceptively simple. In reality, military simulation models are among the most complex simulation models, including hundreds (if not thousands) of modeled elements within each subsystem. If any element within a given federate needs the state information from a modeled element within another federate, then that requested state information will be broadcasted to all other federates. That is, as currently specified, the wrapper to a given federate cannot customize the information that it sends to another federate.

Assume that the number of federates is N, which can become large for complex models. Each federate must establish a link to the remaining N-1 federates. Since the communication between two federates must be two-way, it is clear that the federation of N federates will require N(N-1) real-time directed communication channels. Given the amount of real-time state information that a given federate must broadcast to every other federate within the federation and the number of real-time communication links the RTI must address, it is obvious that considerable communication bandwidth must be provided in order to execute a real-time, distributed simulation.

Some might argue that not all of these communication links will actually be exercised. However, if we are to be selective in specifying which federations will receive information from each federate, then the RTI must provide a messaging service among the federates. That is, one federate must be able to send a specific message to another addressable federate. This does not appear to be the intent of the RTI. Rather the RTI behaves more as a communication bus upon which each federate places its current state information while the other federates listen to the communication bus in order to obtain state updates to external federates whose state impacts their internal operation. Such a design might be efficient if the federates were all situated at a single location and a physical structure that resembles a communication bus could be established. The goal, however, is to have the federates execute at distinct locations. In this case, mimicing the communication bus requires the RTI to send every federate state update to every other federate.

At the 1998 Winter Simulation Conference, others advocated the use of information filtering techniques in order to customize the information that one federate sends to another. (See Murphy and Aswegan [1998]). Again such an approach will require the RTI to send a specific message from the transmitting federate to a specified recipient federate. In order to adopt this approach, significant modifications to the RTI will be required. In effect, one must establish direct (peer-to-peer) communication links among the pairs of federates, an approach that appears to undermine the original HLA concept of providing a single central communication bus.

Another difficulty arises when one attempts to join confederates into a single composite confederate. Consider the two confederations illustrated in Figure 5. The two federations are distinct from each other, and there is no means by which the two federations can communicate. In order to form a single composite federation, several additional links must be added among these federates as shown in Figure 6.

If one views each confederation in Figure 5 as a singular subsystem, the usual desire is to view the composite union of these subsystems as a system of systems. In Figure 6, we observe that the two original federations have been integrated into a single flat closed structure that does not have any hierarchical form. In fact, any information with respect to the prior existence of two distinct confederations is lost. The HLA is incapable of capturing and exploiting the system-of-systems nature for most complex systems. Given this limitation, the HLA structure cannot provide significant capabilities for modeling command and control structures. The need to model command and control structures within the military is obvious.

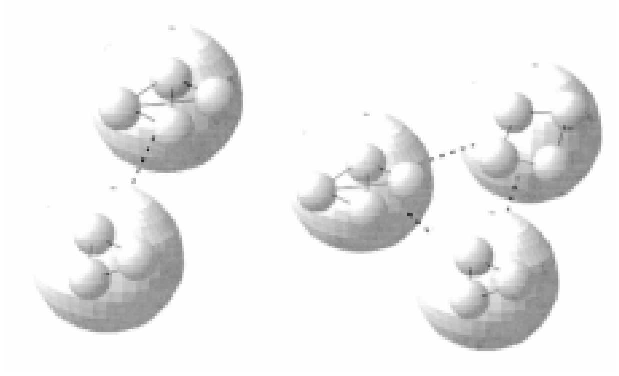

Figure 5: Two Distinct Federations

As we conclude this section, it should be clear the HLA does provide a means to achieve the original goals pertaining to the retention and integration of existing models. The question remains, however: Is it the best architecture? Should HLA become the de facto standard for all distributed simulations for both military and domestic sectors?

2.1 What HLA Cannot Do?

Perhaps the major concern with HLA resides with the framework that it employs to form the composite system. In Figure 6, it is clear that HLA treats each of these federates as peers and establishes communication from each federate to all the other federates. In adopting this framework, HLA completely ignores the fact that most complex systems are actually systems of systems. Furthermore, in adopting a peer-to-peer interaction among the federates, HLA is severely constrained in modeling the command and control structures that are required to manage most complex systems. For military systems, it is obvious that command and control structures are essential. Yet HLA provides no immediate means for assessing the constraints that the command and control structure imposes upon the operation of the system. To effectively model command and control, one must first provide a modeling framework that captures system of systems (or hierarchical) nature of these complex systems.

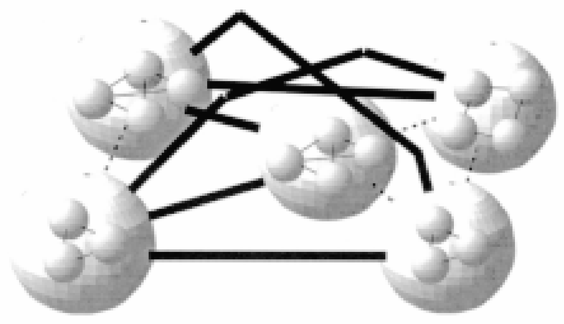

Figure 6: Forming the Composite Federation

In Figure 7, we depict a complex system with its command and control structure given. As with the previous example, we have a set of subsystems at the lowest level, each of which contains a collection of physical elements. Within a given subsystem, the modeled elements can physically interact with each other. As indicated by the rectangular solids in Figure 7, a controller has been included to manage the interactions among the physical objects (indicated by spherical solids) within each subsystem (which the HLA calls a federate). As we attempt to construct a composite system from these subsystems, two types of interactions must be described. The first type of interaction is among the physical objects within the subsystems. It is this type of interaction that HLA addresses in its formation of federates and federations. However, the HLA approach is not the only means that allows the modeled objects within different subsystems to interact with each other.

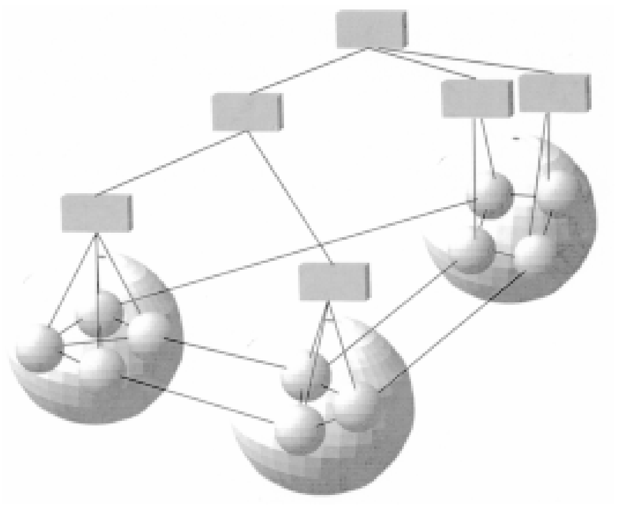

Figure 7: Schematic of a Complex System with Its Command and Control Structure

The Object Modeling Group, Inc. (the same group that adopted the HLA as the standard for distributed simulation) has developed the CORBA standard that will allow programmed objects within the two distinct programs to interact directly with each other. Using the CORBA approach, communication is established directly between the interacting objects. It is obvious that this approach can significantly reduce the amount of communication that must be passed among the objects in real-time. Complete state information for each federate need not be passed to every other federate within the federation. Instead, only the essential information is passed among the interacting objects. In order to employ CORBA, however, two basic approaches within the HLA must be set aside. First, the objects within a federate must be made visible in order to permit them to communicate with other objects in other federates. Second, the notion of a central communication bus must be replaced with object to object communication.

In truth, if one employed the CORBA approach, the entire concept and need for a federate is brought into question. Remember that the concept of a federate is derived from the HLA treating a program as a virtual object. The notion of either a federate or confederation are not system design concepts. A federate represents a subsystem only in the case where the encapsulated program depicts the behavior of a modeled subsystem. A confederation represents only a collection of interacting programs.

The second type of interaction among the subsystems occurs among their controllers. For most complex systems, this is the primary means of interaction. For example, an individual soldier within an infantry group will not likely interact directly with another soldier in a mobile artillery group. However, we can expect that each group's commander will interact with another, either directly or within a command and control structure. In order to model the subsystem interactions within the command and control structure, one must be able to capture the system of systems/ hierarchical nature of these systems. The results of adopting this approach are illustrated in Figure 8.

In Figure 8, we first observe the innermost spheres that define the subsystems (as shown also in Figure 7). Each of these subsystems contains elements that can interact with other elements within the same subsystem as well as in other subsystems. At the next hierarchical level, we create new subsystem objects that include the subsystem with its controller. Each of these newly defined subsystems is managed by a controller at the next hierarchical level. Again, we can encapsulate the newly formed subsystems with their controller in order to form yet another subsystem. This process can be continued until a single object representing the entire composite system is generated.

In defining the above framework for the composite system, not only is its system-of-system nature highlighted, but the multi-resolutional nature of the composite system is also revealed. As we move from the inside spheres toward the outer sphere in Figure 8, an aggregation of information occurs. At the innermost sphere, the objects within a given subsystem interact with each other. At the next level, the controller interacts with each object contained within the controlled subsystem. That same controller also interacts with its supervisory controller. However, it is obvious that the frequency with which the controller interacts with the physical objects under its control is significantly less than the frequency with which it interacts with its supervisor. Furthermore, we can expect that the information that a given controller passes to its supervisor will be less detailed than the information it passes to its subordinate subsystems or objects. In this manner, the multi-resolutional nature for the composite subsystem evolves.

This multi-resolutional approach has additional advantages in a real-time, distributed simulation environment. Not only does this approach more clearly define which objects will be interacting with each other, it also lessens the amount of information that is passed among the interacting objects. Given this, our proposed architecture is more likely to be scalable, allowing one to address extremely large systems.

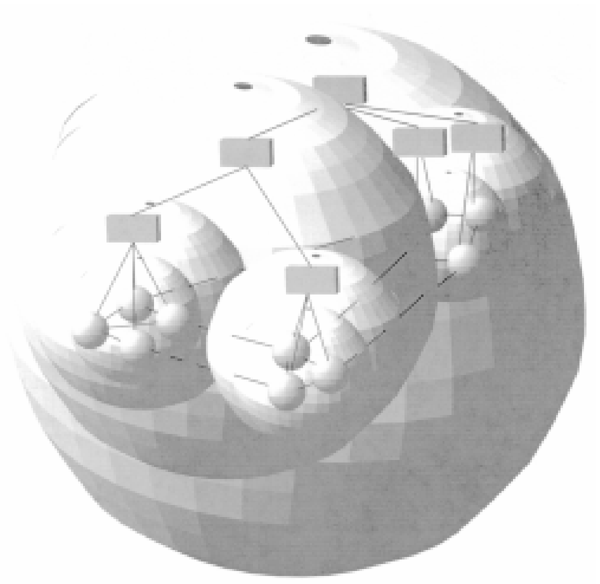

Figure 8: A Schematic for the Systems of Systems Approach

An additional benefit results from adopting the system of systems approach. If we return to Figure 8, any subsystem model associated with a given sphere also represents a stand-alone model that can be executed. Furthermore, if one develops a model using this framework, then the resulting model can be included as a subsystem model within an even larger model. One simply needs to provide a supervisory controller for the subsystem in order to incorporate the subsystem's controller within the overall command and control structure. Next, one uses CORBA to define any interactions among the modeled objects within the new subsystems and the modeled objects within the other subsystems.

The system-of-systems framework also provides guidance on how one should distribute the modeled objects within a distributed simulation environment. In order to minimize cross-platform communication requirements, one should begin at the innermost spheres and attempt to place as many of the subsystems within a given sphere upon the same computing platform. Note that as one moves toward the outer spheres, the communication between the higher level controller and its subsystems will be less frequent. This, in turn, lessens the communication requirements that would result if the subordinate subsystems at this hierarchical level were placed on different platforms.

3 BENEFIT/COST CONSIDERATIONS

Let us now return to original concern of whether the HLA is the ideal solution for meeting the military simulation needs. The fact is that HLA exists and is operating. It certainly allows for prior simulation models to be reused and incorporated within a distributed simulation environment. One might argue that significant cost savings are gained through the ability to reuse the existing simulation models. However, one must also observe that making an existing simulation model HLA-compliant is no trivial feat, nor is it inexpensive.

There are certainly tradeoffs to be considered. Although HLA may accomplish the goals associated with salvaging and integrating existing capabilities, it does not meet the goals associated with system design and management. HLA is not the ideal structure for addressing command and control concerns. Indeed, it is probably unable to consider command and control issues. Also, HLA is not scalable as it is currently implemented. The demands for real-time communication (most of which are unnecessary) that HLA imposes upon the distributed simulation are enormous.

Finally, HLA does not provide a pathway to futuristic applications for simulation. Rather, it salvages existing models, whose future is limited. Given this, should HLA be viewed as the standard for distributed simulation or should it be viewed as a short-term fix until a more comprehensive modeling framework evolves?

4 FUTURISTIC APPLICATIONS FOR SIMULATION

Today, most simulation users employ models to perform off-line simulation analysis. For example, a mission such as Desert Storm was simulated over and over for various strategies in order to determine the best battle plan to be adopted for implementation. However, once a military operation is initiated, the role of simulation is significantly reduced.

Although the use of simulation in an on-line planning and control environment has not been extensively explored, it is clear that simulation can provide phenomenal capabilities in the on-line management of complex systems (see Davis [1998]). In order to illustrate the potential need for on-line simulation, consider the following hypothetical situation. Suppose that North Korea decided to invade South Korea. Unlike Iraq's invasion of Kuwait, where we took several months to plan our response, our response to the North Korean threat would have to be immediate. We can be certain that the military has already analyzed several potential invasion scenarios by the North Koreans and attempted to define the best response for each scenario on an a priori basis. But the actual threat will never correspond exactly to one that was analyzed on an a priori basis.

What the commander must consider is the current state of the threat, the state of his/her forces and the logistic supplies that are available to support the near-term and extended response. At this point, the commander must rely mostly upon his/her intuition. Suppose, however, that the commander could make use of on-line simulation techniques where s/he could test alternative responses even as the current response is being executed. As incoming intelligence on the enemy's force and strength is collected, the implemented battle plan could be refined and tested immediately before it is implemented. In addition, the on-line simulation could provide estimates for logistic needs required to support and sustain the current response.

Through the use of on-line simulation, the logistics side of the house could ascertain its ability to satisfy these needs on a near-term basis and then provide the tactical commander with realistic estimates of the supplies that can be provided. In this manner, the tactical commander can update his/her battle plan in order to maximize the available logistic supplies while guaranteeing his/her force will not extend themselves beyond the point where they can be logistically supported.

HLA will never function in this environment nor will it provide a means to install other DoD initiatives such as simulation-based acquisition. In such initiatives, a given model may be employed for multiple uses within a design, execution and training environment. For example, one might use a physical model of a given piece of equipment in order to establish the tactical benefit that a given feature could provide in battle. The same equipment model, but perhaps at a different level of detail, could be employed in order to develop new battle tactics for using the proposed weapon system. Yet another highly-detailed model of the same equipment might be employed in a trainer. While the acquisition process is in the design phase, we can expect the design will be subject to constant updates. Updating the simulation models at multiple levels of detail in order to reflect the design changes will push object-oriented design practices to the limit. Numerous updates to databases will also be required. For example, as new tactics are developed for the proposed system, its benefits must be determined. A new tactic may also require further design changes which could trigger numerous subsequent simulation analyses. One cannot generate systems with such interactivity capabilities by simply patching together existing models. Such systems must be designed to exploit the latest advances in programming and information technology.

Because HLA was designed to support/salvage past simulation efforts, the resulting models are in turn limited by the capabilities of the models included within the federation. We can assume that many of these models are over a decade old. When we consider the phenomenal advancements that have occurred in distributed object and web-based computing in the last decade, one must question whether old models that cannot exploit these technologies should be retained. We can expect that advancements in computing and networking technologies will continue to expand at an accelerated rate (see Davis, Chen and Brook [1998]). As these future advancements occur, the existing simulation models will become even more outdated. One might conclude that adopting HLA as the distributed simulation standard attempts to draw a line in the sand in order to protect these past models against changing technologies. The problem, however, is that HLA is not simply a line in the sand, but rather a barrier to incorporating advancements in computing and networking technologies.

Perhaps instead of asking whether HLA is the best standard, we should ask ourselves if the military's current simulation capabilities will meet its needs for the next decade. If the answer to this latter question is negative, then one must determine how HLA will constrain the future development. In short, instead of looking to the past and attempting to save existing simulation capabilities, the DoD must look towards meeting its future simulation needs. HLA may not provide a path to that future. If not, then the DoD must seek a better way.

REFERENCES

Davis, W. J. 1998. On-Line Simulation: Need and Evolving Research Requirements. In *Handbook of Simulation*, ed. J. Banks, 465-516. New York: John Wiley and Sons, Inc.

Davis, W. J., X. Chen and A. Brook. 1998. Implementing On-Line Simulation upon the World-Wide Web. *Proc. of the 1998 Winter Simulation Conference*, 87-95.

Murphy, W. S. and Aswegan, G. D. 1998. High Level Architecture Remote Data Filtering. *Proc. of the 1998 Winter Simulation Conference*, 835-840.

US Department of Defense. 1997. High Level Architecture Interface Specification, Version 1.1. Defense Modeling and Simulation Office.

AUTHOR BIOGRAPHIES

WAYNE J. DAVIS is a professor of General Engineering at the University of Illinois @ Urbana-Champaign. His research addresses the distributed intelligent control architectures for complex systems. To support this development, he has developed several new modeling paradigms and online simulation approaches.

GERALD MOELLER has held a variety of positions in Army for the past 28 years in support of weapon systems project development. During this tenure, he developed the Venture Evaluation and Review Technique (VERT) simulation tool, which has been widely used in DoD, DoC and private industry for the past 20 years to analyze project risk. Mr. Moeller was named Army Systems Analyst of the Year for his work in simulation.