Proceedings of the 1996 Winter Simulation Conference
ed. J. M. Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain

A PERFORMANCE ANALYSIS MODEL FOR DISTRIBUTED SIMULATIONS

David B. Cavitt
C. Michael Overstreet
Kurt J. Maly

Department of Computer Science
Old Dominion University
Norfolk, Virginia 23529-0162. U.S.A.

ABSTRACT

Distributed simulation has proved to be a cost effective technique for studying and understanding complex real-world systems. Many distributed simulations need to incorporate hundreds or even thousands of processors, across both local and wide area networks. As the size and complexity of distributed simulations increase so do the demands on the hardware and software resources that provide simulation services, causing performance bottlenecks that limit the effectiveness of distributed simulation. Different abstractions of performance are needed depending on who is analyzing the distributed simulation and for what reason. This paper proposes a framework for identifying the factors affecting performance and provides a technique for associating the performance factors with high-level system metrics that describe the behavior of the physical and logical resources and services used in the design and implementation of distributed simulations. Dynamic and static analysis of the performance information provides feedback on the execution of the simulation and can provide meaningful information as a guide in making decisions about the configuration and control of the available hardware and software resources for distributed simulation exercises.

1 INTRODUCTION

Distributed simulation is an effective technique for systems studies in research, acquisition, and training environments. For many applications, distributed simulations consist of using large numbers of workstations integrated into local-area and wide-area networks. To improve the benefit of distributing a simulation's computations and run-time environment it is necessary to understand the simulation's performance characteristics in the context of the distributed environment. The concept of distributed simulation performance is multifaceted and not easily characterized. Concurrent execution of simulation models and replicated resources and services are based on abstract models of computation, and performance characterizations of this system can be difficult to understand. The need for greater realism results in demand for resources that often exceeds the nominal capacity (both spatially and temporally) of the workstations currently used as simulation engines. When assessing distributed simulation performance, the goal of simulation speedup dominates most of the literature (Fujimoto 1990, Carothers and Fujimoto 1994, Falsafi and Wood 1994). Number of entities per workstation (where an entity is defined as some simulated real-world object) is also used as the singular metric of performance in Vrablik and Richardson (1994) and the ModSAF 1.4 Reverse Engineering Report (1995). Speedup and entities per workstation however are not sufficient for a full characterization of distributed simulation performance. This is especially true when distributed simulation is used in training and other man-in-loop environments. Performance information relates not just to the speed of computation, but to the efficiency and effectiveness of the simulation in utilizing the many shared resources and services within the distributed environment. Performance information characterizing concurrency, scaling, replication, and coordination can help characterize the simulation's ability to meet the many goals of the simulation study.

Further complicating the notion of performance, the utility of information depends on the needs of the intended audience. Different phases of the simulation life-cycle require people in different roles, each requiring

different kinds of information relating performance with simulation goals. The roles include that of model developers and simulation programmers, configuration and experimentation planners, and system analysts or program managers. Performance information gathered and used by these people during the various phases of the simulation life cycle differentiates between the distributed simulation's different physical and logical components (i.e., hardware, O/S, application, models, experimentation). A systems analyst or project manager will generally be interested in high-level performance information on the simulation's capabilities, utilizations, and bottlenecks as they relate to the simulation study's goals. A model developer or programmer can use some of the same high-level information, but will most likely need additional measured data to derive lower-level information. Configuration and experimentation planners also need high-level information that allows them to assess the performance of the simulation. With the recognition that the distributed environment imposes certain constraints on the simulation's processing requirements, configuration and experimentation planners must have both low-level and high-level performance information that allows them to understand the affects of computer, network, and other architectural components.

Typical hardware performance information in distributed simulations, like CPU and network utilization, are easy to identify and measure using traditional tools. Obtaining performance information specific and meaningful to each of the life-cycle roles requires alternative methods, deriving the information from the composition of data characterizing the local run-time performance of individual services and resources, and global data on the structure and performance of the entire distributed simulation environment. The perception of both local detail and global structure provides both coarse and fine-grained views of distributed simulation performance. The performance information is represented at a level of detail that is sufficient for both the design and development, and the configuration and execution of distributed simulations.

Traditionally, most performance measurement and analysis methods of distributed simulations have been application specific. The facilities for measuring and analyzing the performance are generally tightly-coupled to the structure of the simulation and are intrusive during run-time execution, adversely impacting simulation performance measurements. System metrics are typically based on a set of criteria that indicate achievement or not of a particular performance goal, but do not articulate the reasons for the observed performance and behavior. Speedup or the number of simulated entities per workstation are examples of such metrics. The number of components that make up a distributed simulation and the complexity of interaction among the components make identifying all the performance factors a difficult task. A well-defined performance model establishes a relationship among these factors and the constructs and abstractions used in the simulation's design and implementation. The fundamental resource and service components of simulations, model characteristics, and distributed systems, provide a basis for identifying the performance model abstraction. The level the performance information is presented at is determined by the role and intended use of the information. The goal is to provide performance information to different decision makers - in terms of choices available to that individual - which could for example improve performance or indicate potentials for additional model actions. The intended use of the performance information defines what metrics are necessary to develop an understanding of the simulation's performance.

This paper discusses the preliminary concept and design of a distributed simulation performance model. The model describes the framework we are using to guide the process of identifying metrics that provide a specification of distributed simulation performance. The key components of the performance analysis model are still under development. An initial implementation of the model has provided data for evaluating the processing overhead of the performance measurements and verified the utility of aggregating and presenting meaningful performance information to decision makers. Section 2 defines the performance model abstraction and the semantics of the performance information as it relates to the physical and logical components of distributed simulation. Section 3 discusses the use of the performance model, the presentation of performance information, and gives examples of its use with ModSAF, a distributed wargaming simulation built by Loral Advanced Distributed Simulation and used for military training and doctrine development. Section 4 summarizes the proposed model.

2 PERFORMANCE ANALYSIS MODEL

Performance information typically has both space and time dimensions and can be considered the specification of the simulation's execution. To specify the execution we must describe where and when in the distributed simulation certain behaviors exist, which correlates with the space and time dimensions. The performance model represents the space dimension by using three sets of logical components that are a part of distributed simulations; distributed resources, simulation services, and simulation model characteristics. Distributed resources are considered objects or entities that utilize

some key characteristics of distributed systems (such as concurrency). Simulation services are those logical components used in the architecture of present day discrete event simulations (such as event schedulers). The model characteristics are those components used to define and represent a simulation model (such as the entity attributes). Each set of components makes up a single dimension and when combined make up a three dimensional discrete space of factors effecting the performance of distributed simulations. Each factor is a function of one component from each dimension. Performance information associated with each factor is derived during the simulation exercise from the combined effect of each of the components. Simulation metrics establish a mapping between the performance factors defined by the model and the simulation's physical and logical resources and services. The performance information associated with each factor can consist of many metrics. A fourth dimension, time, represents the changing phases of simulation performance as time passes during the simulation exercise. Figure 1 is a graphical representation of the space and time dimensions of the performance model.
Each shaded cube represents all possible factors that affect the simulation's performance at any instant in time. This representation attempts to define a framework from which a full spectrum of simulation metrics can be defined and used for performance analysis.

2.1 Space and Time Components

As described above, the three dimensions representing the space of performance factors consist of a set of distributed simulation components. The components represent the distributed simulation's resources, services, and model characteristics and define where and why in the simulation's execution space a certain behavior or performance factor is observed. One of the primary purposes of developing this performance model is to present a unified framework for describing the performance of distributed simulation. The challenging aspect of defining the model is to make sure the enumerated sets of components comprising the

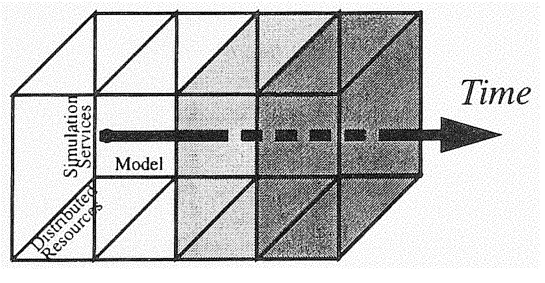

Figure 1: Representation of Performance Factors

performance model encompass the full spectrum of physical and logical components used in the distributed simulation's design and implementation. Figure 2 shows the three dimensions and the corresponding components used to define the model's performance factors.

The discrete space representation of the performance model provides a well-defined abstraction of logical components affecting the performance of distributed simulations. An important point about the representation is that since each dimension in space represents an enumerated set of components, no notion of magnitude or value is associated with the dimension (as is done in vector analysis). The location of each component in its dimension is merely a recognition of the existence of that component as a contributing factor to simulation performance. The discrete space representation results in a set of sixty potential performance factors, each factor being a function of the combined effect of distributed resources, simulation services, and model characteristics. Assessing simulation performance at any instant in time can be obtained by analyzing the three-dimensional discrete space model. A broader understanding of simulation performance can be gained by examining the changes to the discrete space model throughout the simulation exercise. Thus the time dimension is included as part of the model, relating the run-time characteristics of the distributed simulation to changes in performance. The resulting discrete space-time representation provides a high-level abstraction for identifying and communicating the spatial and temporal aspects of distributed simulation that effect performance.

The model's set of distributed resource components are derived from properties of distributed systems: concurrency, scaling, replication, and coordination. To benefit from the use of workstations in current network topologies, distributed simulation environments must be designed to incorporate, exploit, and compensate for these properties. Although these components interact, each can be considered separately to achieve an overall design objective. Concurrency, the parallel execution of simulation and model components, is an artifact of physically separate computers providing simulation resources and services (e.g., the coexistence of simulation entities). Scaling entails increasing the number of shared or replicated resources, services, and simulation entities with a goal of having no single simulation resource or service being in restricted supply. Replication, the existence of multiple copies of simulation data, is a mechanism to increase reliability and fault tolerance (a qualitative goal), and achieve better performance as it relates to higher availability of shared resources and services. Coordination synchronizes the timing and ordering of simulation events including the proper ordering of human intervention with the

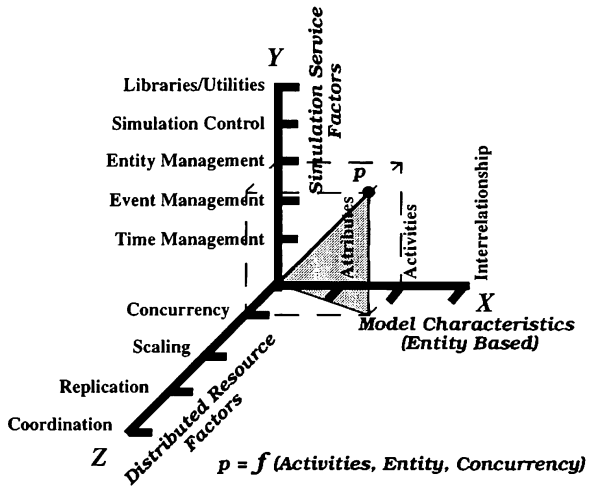

Figure 2: Performance Factors Representation

simulation (where applicable). Each of these components is used to define performance factors of distributed simulation.

Another dimension of the performance factor space deals with simulation service components. The model's service components are the functional components that are part of discrete event simulations. The services include the management of events and event schedulers, entity management, simulation administration and control, and support routines and libraries (such as random number and random variate routines). Distributed simulations generally have multiple schedulers executing concurrently on each of the computers participating in the simulation. The schedulers must invoke events in some coordinated or synchronized manner, assuring either correct program behavior or the recovery from errors induced by uncoordinated activity. The schedulers should ideally also execute efficiently, independent of the number of events that must be serviced as the size of the simulation experiment is scaled up or down. The aggregate cost of maintaining and accessing multiple and replicated events lists and other data structures can be significant and is considered along with each of the distributed resource components. Entity management in distributed simulation is affected by concurrency, scaling, replication, and coordination. The simultaneous creation, execution, and deletion of entities on separate computers relates the concurrency resources to the entity management services and the ability to increase the number of entities in the simulation relates the scaling component. Similarly, the replication and coordination requirements of the simulation affect the management of the simulation entities.

The control of distributed simulations (e.g., start-up, termination, cleanup operations) also affects run-time behavior and must be considered when assessing simulation performance. Additionally, for some simulations graphical displays and high levels of user interaction and intervention can be intrusive to the simulation and its impact on performance must be understood. Lastly, the library and utilities associated with distributed simulations are diverse and application dependent. However, the affect on performance can be dramatic depending on the amount of concurrency and replication, and whether or not the data associated with the libraries and utilities are shared.

The service components, when considered with the distributed resource components, define where in the distributed simulation's execution space a certain performance factor can be observed. The simulation's model characteristics make up a third dimension in the performance factor space and are combined with the resource/service pairs of components to define sources of performance information. The model characteristics are defined by entity attributes that describe some real-world object, entity activities that are the modeled behaviors of the entity, and entity interrelationships that are defined as any dependencies, interactions, or shared behaviors that exist among entities. The concurrent simulation of entities can cause resource and service contention, especially as the number of entities is scaled up. When an interrelationship exists between entities the replication and coordination components can significantly affect performance.

2.2 Simulation Metrics

The performance information associated with each factor defined in the space-time representation can be derived from run-time measurements of the distributed simulation and from static code analysis of the simulation code. Simulation metrics map the components of the space-time representation to physical and logical services and resources used in the distributed simulation environment (i.e., CPUs, network interfaces, event schedulers). The metrics become a specification of the simulation's execution expressed in terms of these services and resources; time-varying functions that characterize the distributed simulation's performance. The domain of metrics includes time, rate, utilization, reliability, and availability for each of the services and resources that contribute to a performance factor. The time metric domain is related to the throughput or responsiveness of the simulation in providing the service. The rate metric domain is related to the productivity or the frequency with which the simulation provides the service. The utilization metric domain is related to the utilization or the amount of the service the simulation uses or needs. The reliability metric domain relates to the

correctness with which the simulation provides the service and its ability to meet the deadlines when providing the service. The availability metric domain relates to the simulation's ability to provide the service as required. These metric domains are applicable to each computer participating in the simulation and provide information on individual machines' performance. The same domains are used to define a global view, relating the concurrency, scaling, replication, and coordination components of the performance model with the overall progress and performance of the distributed simulation

Table 1 shows one possible set of simulation metrics that map the entity management services of a distributed simulation with the distributed resource components. The model characteristics are implicit within the definition of the metrics. The column labeled "Autonomous Simulation" shows metrics for the service as that metric pertains to each individual computer participating in the simulation exercise. The metrics relating the distributed resource components to the entity management service are listed in their respective columns and provide a global view of performance. The metrics shown primarily address the processing times associated with simulation entities. The processing time is the simulation work cycles spent executing entity behaviors and activities as well as the modification of entity attributes. The term "node" refers to any specified computer participating in the distributed simulation exercise. The time and rate domains are generally inversely related, the former giving the average amount of time the service takes for each entity type and the latter specifying the average frequency that the service is provided. The global metrics specify the performance across all computers participating in the simulation exercise. The individual metrics provide a more fine-grained view of the performance of specific computers. The reliability metrics assume certain deadlines or thresholds exist for processing entities and the metrics associated with the scaling component of the distributed resources assumes that certain nominal capacities exist. These deadlines and capacities could be defined by simulation requirements, previous empirical studies, or set arbitrarily.

The sample metrics present performance information at a consistent level of detail and are at a sufficiently high-level of abstraction, should allow an intuitive understanding of performance as it relates to the resources and services that are part of distributed simulation. Low-level simulation metrics such as transmission rate for a network interface or the average search time for getting the next event off of the event list are not presented. Low-level performance data, however, might be needed for deriving higher-level performance information.

The physical resources that make up a distributed simulation environment include CPUs, memory and buses, disk drives, network interfaces and transmission media. The operating system functions and interfaces provide access to these resources. The simulation services that request the resources include event schedulers, event-list management routines, clock and timer routines, random number generators, user interface and input/output routines, database routines, and others. The measurement of these physical and logical resources contributes data used to derive simulation metrics. The data can include count data, frequency data, and trace data. Count and frequency data of the physical and logical resources is considered low-level in terms of information content and it is used to derive higher-level simulation metrics that are defined by the performance model. Trace data may be low-level or high-level (containing actual measures of performance). For many distributed simulations, especially long running ones, the volume of measured data can be prohibitive to store, so processing low-level data into high-level trace information can have a distinct advantage (smaller storage requirements and reduced network utilization). The disadvantage of saving only high-level trace data is that it is difficult or impossible to recreate the analysis since content is lost once the data are processed into higher-level performance information.

3 USE OF MODEL

A primary concern of distributed simulation exercise and configuration planners is how to best utilize the available hardware and software to achieve the goals of the simulation exercise or experiment. The performance model provides a framework for specifying the simulation's performance at a level of abstraction that is suitable for making utilization decisions. Once the simulation is capable of providing the raw performance data, the metrics can be derived and the associated performance information analyzed to identify major performance problems. The global simulation metrics are used to observe overall performance. If a bottleneck exists somewhere, a finer-grained analysis can be done by isolating the components of the performance model that are contributing to the bottleneck, and analyzing the individual simulation metrics associated with each factor. After observing and analyzing the performance information, any changes in system configuration can be formulated, and further analysis done to assess the impact on performance. This kind of analysis can be an iterative process, repeatedly looking at performance information to identify exact reasons for performance problems. Why and where there are problems in the simulation's execution is just one aspect of using the performance

Table 1: Mapping of Service and Resource Components to Simulation Metrics

Simulation Service	Metric Domain	Autonomous Simulation (Individual Metric)	Distributed Resources (Global Metrics)			
			Concurrency	Scaling	Replication	Coordination
Entity Management	Time	Avg. Time To Process An Entity	Avg. Time To Process An Entity Per Node	Est. Δ in Time to Process An Entity Per Node With Entity Addition/Deletion Variance in Time To Process An Entity Per Node	Avg. Time To Process Replication Per Node	Avg. Time To Process Entity Coordination Tasks
	Rate	Avg. Number Of Entities Processed Per Unit Time	Avg. Number of Entities Processed Per Unit Time Per Node	Est. Δ in Number of Entities Processed Per Unit Time w/Entity Addition/Deletion Variance in Number of Entities Processed Per Unit Time	Avg. Number of Replications Processed Per Unit Time	Avg. Number of Entity Coordination Tasks Per Unit Time
	Utilization	Total Number of Entities Processed	Avg. Number of Entities Processed Per Node	Variance in Number of Entities Processed Per Node	Total Number of Replicated Entities	Total Number of Entity Interactions
	Reliability	Percentage of Total Run-Time Entity Processing Misses Deadlines	Avg. Number of Entities Missing Deadlines Per Node	Est. Δ in Number of Entity Additions/ Deletions To Deadline Variance in Percentage of Total Run-Time Entity Processing Misses Deadlines	Percentage of Total Run-Time Replication Processing Misses Deadlines	Avg. Number of Entity Interactions Using Stale Data
	Availability	Avg. Elapsed Time Entity Spends Waiting To Be Processed. Percentage Of Total Run-Time Spent Processing Entities.	Concurrency Index (Variance In The Number of Entities Processed Per Unit Time For All Nodes)	Est. Allowable Number of Entity Additions/Deletions	Replication Index (Variance In The Number of Replications Processed Per Unit Time For All Nodes)	Percentage of Total Run-Time Spent Processing Entity Coordination Tasks

model. In what phase of execution does the simulation's performance become limiting? This question is answered by analyzing the simulation metrics over time. The performance model provides the framework for analyzing independently, the different phases of the simulation's execution.

As an example of how the model can be used, consider an implementation of Loral's ModSAF, a Distributed Interactive Simulation (DIS) used for combat training and military doctrine development. A primary goal of ModSAF is to replicate the behavior of simulated military units, vehicles, and weapons systems to a level of realism sufficient for training military personnel (ModSAF Software Architecture Design and Overview Document 1995). Meeting this goal requires the simulation to heavily populate the virtual battlefield, simulating entity on entity combat engagements in an autonomous manner while allowing a user to maintain supervisory control of a large number of military units or individual vehicles.

Examine the availability metric at the bottom of Table 1. The proposed metric estimates the allowable number of entities that can be added to the existing simulation workload. The performance analysis model identifies this metric by the interdependencies of the entity management component of the simulation services (schedular), a scalability component of the distributed systems resources (time available for additional processing), and the entity attributes, activities, and interactions characterizing simulation models. An initial version of the performance model has been implemented by instrumenting ModSAF. During an experiment, data points were measured and derived from three aspects of the execution space of the DIS simulation; 1) the number of vehicles simulated on a single workstation, 2) the percentage of vehicle processing time spent handling the DIS interprocessor communication data packets called Protocol Data Unit (PDUs), and 3) the amount of time available for additional vehicle processing, referred to as slack, within the time constraints defined for the update rates of vehicle state data (Foster et al. 1994). Figure 3 shows a plot of the data. Note that the curve has not been smoothed, revealing the variations in required processing times dependent upon vehicle behavior.

The availability metric provides an estimate on the number of vehicles that can be added to a simulation exercise while still achieving acceptable simulation performance. This metric may be a linear or non-linear function of the slack time and the per vehicle processing time. Referring to Figure 3, we can see with 25 simulated vehicles approximately 140 milliseconds of slack exists (given a 2 Hz update rate for vehicle state). Assuming a simple linear function for calculating the availability metric, then a vehicle processing time of 14.4

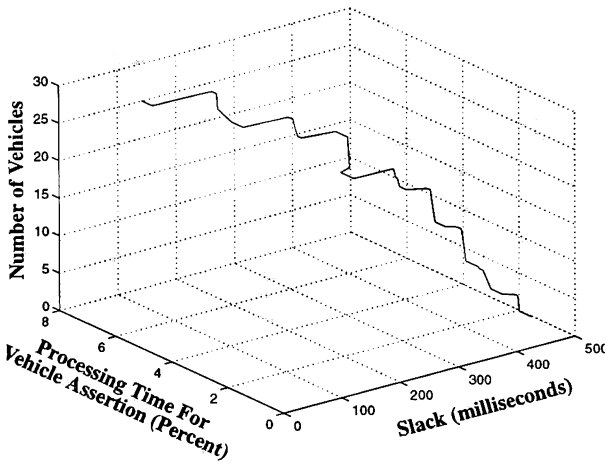

Figure 3: Relationship Among Number of Vehicles, Vehicle Assertion Processing, and Slack Available for Vehicle Processing

milliseconds per update means it is possible to add approximately 9 more vehicles and still meet the required 2 Hz update rate. This information could be derived at run-time and used for example by an analyst or decision maker who requires a greater number of vehicles to enhance the realism of a particular ModSAF simulation exercise.

This example serves to illustrate how a performance model that presents performance information at a high-level of abstraction can be used to guide analysts and configuration personnel in assessing distributed simulation performance. The implementation shows the feasibility of data collection and no sophisticated understanding was required of the simulation's hardware or software processing requirements (i.e., CPU or function timing data) nor any specialized tools required.

4 SUMMARY

The need to assess the limiting factors in the performance of distributed simulation lends itself to the development of a performance analysis model. Development of the model requires the identification of performance problems by understanding why the simulation performs the way it does, where the bottlenecks in performance occur, and when the effective performance of the distributed simulation is no longer adequate. This paper has discussed a performance model for distributed simulation. The need to specify performance using a high-level of abstraction, makes the proposed model suitable for performance assessment during all phases of the simulation life cycle. The model components are defined using a service/resource paradigm that maps the actual physical and logical components used in distributed simulations to the resource and service abstractions of the performance model. The model defines metrics that characterize distributed simulation, specifies what aspects of the simulation and distributed environment are to be observed, the analysis to be performed, and clarifies the semantics and use of the analysis results. Examples of the use of the model, while somewhat artificial in terms of scenario, were representative of the utility of a high-level abstraction for distributed simulation analysis.

An initial implementation of the performance analysis model has demonstrated the ability to collect performance data at a low level of intrusiveness, and can be used to derive metrics which help decision makers during the design of ModSAF training exercises. Future work will involve the refinement and validation of the performance model components, a complete definition of simulation metrics to fully map the set of service and resource components with the model characteristics, and the design and implementation of statistical techniques for presenting meaningful performance information usable throughout the simulation life-cycle. Attempts will be made to validate the use of the performance model during actual simulation exercises using the ModSAF simulation. Areas for future research include exploring the use of static code analysis for identifying resource and service requirements of the simulation as well as potential interactions and dependencies between simulation entities. Open research issues also include the design, use, and assessment of predictive techniques to guide decisions on how to better utilize hardware and software resources used in distributed simulation.

ACKNOWLEDGMENTS

Support for this research was funded by BMH Associates, Inc., Norfolk, VA, under grant number BMH-01N61339-95-C-0030.

REFERENCES

Carothers, C. D., and R. M. Fujimoto. 1994. Effect of Communication Overheads on Time Warp Performance: An Experiment Study. In *Proceedings of the 8th Workshop on Parallel and Distributed Simulation*, ed. D. K. Arvind, R. Bagrodia, Jason Yi-Bing Lin, 118-125, IEEE Computer Society Press, Los Alamitos, California.

Falsafi, B., and D. A. Wood. 1994. Cost/Performance of a Parallel Computer Simulator. In *Proceedings of the 8th Workshop on Parallel and Distributed Simulation*, ed. D. K. Arvind, R. Bagrodia, Jason Yi-Bing Lin, 173-182, IEEE Computer Society Press, Los Alamitos, California.

Foster, L., P. Maassel, and D. McBride. 1994. The Characterization of Entity State Error and Update Rate for Distributed Interactive Simulation. In *Proceedings of the 11th DIS Workshop on Standards for the Interoperability of Distributed Simulations*, September 1994, 61-73, Institute for Simulation & Training, Orlando, Florida.

Fujimoto, R.M. 1990. Performance of Time Warp Under Synthetic Workloads. In *Proceedings of the SCS Multiconference on Distributed Simulation*, ed. David Nicol, 23-28. Society for Computer Simulation, San Diego, California.

ModSAF 1.4 Reverse Engineering Report. 1995. Technical Report. Applied Research Laboratories, University of Texas at Austin.

ModSAF Software Architecture Design and Overview Document (SADOD) 1995. LADS Document Number 94070v1.0. Loral Advanced Distributed Simulation, Cambridge, Massachussetts.

Vrablik, R., and W. Richardson. 1994. Benchmarking and Optimization of ModSAF. In *Modular Semi-Automated Forces: Recent and Historical Publications*, LADS Document No. 94007 v.1.0, 141-147. Loral Advanced Distributed Simulation, Cambridge, Massachussetts.

AUTHOR BIOGRAPHIES

DAVID B. CAVITT is a doctoral student at Old Dominion University, Norfolk, Virginia. He received a BS degree in Computer Science at Old Dominion University. He has 9 years of experience in the use and development of simulations for military and engineering applications. His research interests include modeling and simulation, performance analysis, and distributed systems. David Cavitt is a member of ACM, and IEEE CS.

C. MICHAEL OVERSTREET is an Associate Professor of Computer Science at Old Dominion University. He is immediate past chair of the Special Interest Group in Simulation (SIGSIM) of the ACM. He received his B.S. from the University of Tennessee in 1966, an M.S. from Idaho State University in 1968, and an M.S. and Ph.D. from Virginia Polytechnic Institute and State University in 1975 and 1982 respectively. He has been a visiting research faculty member at the Kyushu Institute of Technology in Japan. His current research interests include model specification and analysis, distributed simulation, high performance networking, support for interactive instruction, and static code analysis in support of software maintenance tasks. He is currently a principal investigator in tasks funded by ARPA, ICASE at NASA Langley, and the National Science Foundation. Dr. Overstreet is a member of ACM, and IEEE CS.

KURT J. MALY received the Dipl. Ing. degree from the Technical University of Vienna, Austria, and the M.S. and Ph.D. Degrees from the Courant Institute of Mathematical Sciences, New York University, New York, NY. He is Kaufman Professor and Chair of Computer Science at Old Dominion University, Norfolk, VA. Before that, he was at the University of Minnesota, both as faculty member and Chair. He also is Visiting Professor at Chengdu University of Science and Technology, People's Republic of China and is Honorary Professor at Hefei University of Technology, PRC. He was a member of Board of the Microelectronic and Information Sciences Center, Minneapolis. His research interests include modeling and simulation, very high-performance networks protocols, reliability, interactive multimedia remote instruction, Internet resource access, and software maintenance. His research has been supported by DARPA, NSF, NASA, CIT, ARPA and the U.S. Navy among others. Dr. Maly is a member of the IEEE Computer Society and the Association for Computing Machinery.