Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

ENABLING 1,000,000-ENTITY SIMULATIONS ON DISTRIBUTED LINUX CLUSTERS

Gene Wagenbreth
Ke-Thia Yao
Dan M. Davis
Robert F. Lucas

Information Sciences Institute,
4676 Admiralty Way
University of Southern California
Marina del Rey, CA 90292, U.S.A.

Thomas D. Gottschalk

Center for Advanced Computing Research
1200 E. California Boulevard
California Institute of Technology
Pasadena, CA, 91125, U.S.A.

ABSTRACT

The Information Sciences Institute and Caltech are enabling USJFCOM and the Institute for Defense Analyses to conduct entity-level simulation experiments using hundreds of distributed computer nodes on Linux Clusters as a vehicle for simulating millions of JSAF entities. Included below is the experience with the design and implementation of the code that increased scalability, thereby enabling two orders of magnitude growth and the effective use of DoD high-end computers. A typical JSAF experiment generates several terabytes of logged data, which is queried in near-real-time and for months afterward. The amount of logged data and the desired database query performance mandated the redesign of the original logger system's monolithic database, making it distributed and incorporating several advanced concepts. System procedures and practices were established to reliably execute the global-scale simulations, effectively operate the distributed computers, efficiently process and store terabytes of data, and provide straightforward access to the data by analysts.

1 INTRODUCTION

As background, the authors set out a brief review of the entity-level simulations used by the Joint Forces Command (JFCOM) and others in the Department of Defense. The need for increased scope and resolution is covered, which led to the consideration of high performance computing (HPC). They will then cover the architectural constraints, the implemented designs, and performance testing and analysis. This will set the stage for the major thrust of the paper: the experiences with operations and data management. The paper concludes with a look at future work and suggests some conclusions that may be drawn at this time.

The United States is facing an entirely new environment of politics, economics, threats and conflict settings. (Barnett 2004). This is forcing a deep and broad re-evaluation of its defensive posture. In looking to the future, the DoD has a vested interest in being able to simulate more than one million vehicles, each with sophisticated, independent behaviors. This is driven by the government's needs to effectively use new computer and communications technology in the U.S. defense organizations (Cebrowski 1998) and simulate more complex human functions (Ceranowicz 2004) in technically diverse situations (Sanne 1999). The U.S. Department of Defense (DoD) has begun a series of experiments to model and simulate the complexities of urban environments. In support of their mission, analysts need to conduct interactive experiments with entity-level simulations, using programs such as the Semi-Automated Forces (SAF) family used for years by the DoD (Ceranowicz 2002).

This needs to be done at a scale and level of resolution adequate for modeling the complexities of military operations in urban areas. All of this leads to the analysts' requirement of simulations of at least 1,000,000 vehicles or entities on a global-scale terrain with high-resolution insets. Experimenters using large numbers of Linux PCs distributed across a local area network (LAN) found that net communications constraints limited the analysts to tens of thousands of vehicles, about two orders of magnitude fewer vehicles than their needs. This paper addresses the benefits of the successful application of computational science and parallel computing on High End Computers to this situation. By extension, it lays out a template for those with similar simulation needs, who can make beneficial use of such computers, one group of which is often called Scalable Parallel Processors (SPPs). JFCOM used assets of the High Performance Modernization Program (HPCMP.)

While there are many Forces Modeling and Simulation (FMS) approaches that are currently in use, experimentation at the entity level provides some very attractive features, both for training and for analysis. Making these simulations so that the user can directly involve humans, *i.e.* Human-in-the-Loop (HITL), additionally augments the DoD's ability to assess true impacts on personnel and procedures. (Ben-Ari 1998) There are several new methods to modeling human behavior (Hill 2000). While these require significant independent research (vanLent 1998), they also require significant additional computing power. Current PC capability does not allow the analyst to conduct these experiments at the scale and level of resolution necessary and much of the FMS community currently reports not using High Performance Computing (HPC). (Davis 2004) These constraints and disinclinations have also been found in other varieties of simulation. (Kaufman 1993)

In the present case, JFCOM's newfound emphasis on civilian entities, called "clutter," has extended the horizons of entity-count requirements by approximately two orders of magnitude. In any urban setting, the number of civilian vehicles will easily outnumber the combat vehicles by a factor of ten, and more likely, by a factor of 100. Trying to assess the utility of sensors and the efficacy of intelligence analysis in discriminating the combatants from the civilians will putatively be insufficiently served by a simulation that is limited to a few thousand vehicles total.

The current work on the authors' Joint Experimentation on Scalable Parallel Processors (JESPP) Linux clusters enabled the successful simulation of the required 1,000,000 entities. Software implementations stressing efficient internode communications were necessary to achieve the desired scalability. Further, this is a challenge that the authors assert may only be amenable to meta-computing across widely dispersed and heterogeneous parallel computer assets (Foster 1997). One major advance was the design of both the "Tree" and the "Mesh" software routers to efficiently route information between simulators on local and wide area networks.

The work reported on in this paper is based on the earlier work funded by DARPA in the mid nineties, headed by Paul Messina at Caltech (Messina 1997). The Synthetic Forces Express project (SF Express) was conceived and initiated to explore the utility of SPPs as a solution to the communications bottlenecks that were then being experienced by one of the conventional SAFs, ModSAF. The SF Express charter was to demonstrate the capability of practically hosting a scalable communications architecture on multiple SPPs to simulate 50K vehicles: an order-of-magnitude increase over the size of an earlier major simulation, Synthetic Theater of War–Europe (STOW-E).

The professionals from the SF Express effort, many of whom are on the JESPP staff, were able to mount a simulation run, with more than 100,000 individually simulated vehicles in March of 1998. The runs used several different types of SPPs at nine separate sites spanning seven time zones. These sites were linked by a variety of wide-area networks. (Brunett 1998)

That work was built on the DIS standard utilized by the SAFs at that time. That standard was replaced by the HLA/RTI standard that was purportedly more scalable, but several years of use has shown the clear limits of this new approach. This has not prevented some experimenters from getting very good results while simulating approximately 30,000 entities (Ceranowicz 2002). These new standards and additional requirements have driven the development of the two new router designs, Mesh and Tree Routers.

1.1 JSAF

The Joint SemiAutomated Forces simulation suite is used by the US Joint Forces command in its experimentation efforts. JSAF runs on a network of processors, which communicate via a local or wide area network. Communication is implemented with High Level Architecture (HLA) and a custom version of Run-Time Infrastructure (RTI) software version RTI-s. A run is implemented as a federation of simulators or clients. Multiple clients in addition to JSAF are typically included in a simulation. The typical user-interface is a map visualization or a three-dimensional, rendered view as are shown in Figure 1.

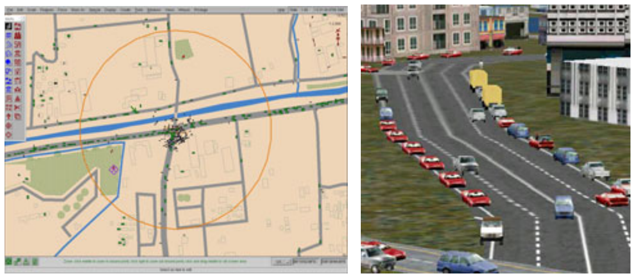

Figure 1: Plan View and 3D Rendered Displays from a SAF

HLA and RTI use the publish/subscribe model for communication between processors. Typically, these processors are in relatively powerful PCs using the Linux operating system. A data item is associated with an interest set. Each JSAF instance dynamically subscribes to ranges of interest. A JSAF may be interested in, for example, a geographic area or a range of radio frequencies. When a data item is published, the RTI must send it to all interested clients.

A typical JSAF run simulated a few thousand entities using a few workstations on a LAN. A simple broadcast of all data to all nodes is sufficient for this size simulation. The RTI on each node discarded data that was not of interest to each receiving node. Broadcast is not sufficient when the simulation is extended to tens of thousands of entities and scores of workstations. UDP multicast was implemented to replace the simple broadcast. Each simulator received only the data to which it has subscribed, *i.e.* in which it has a stated interest.

Operational imperatives drive experimental designs that now require further expansions of JSAF capabilities such as more entities, more complexity, larger geographic area, multiple resolution terrain, and more complex environments. The most readily available source of increased compute power, up to one or more orders of magnitude, is the capability presented by Scalable Parallel Processors. In the JESPP project, JSAF was implemented on multiple Linux clusters, using hundreds of processors on each cluster. Future runs will require thousands of processors on multiple clusters. The obstacle to using these resources effectively was the poor scaling of the original inter-node communication.

Both Tree and Mesh software routers were implemented on individual nodes within the local mesh network that also included all of the client simulators. Each simulator is connected to only one router. Routers are connected to multiple clients and multiple routers. Two types of information are present: data along with interest description and the current interest state of each client. The interest state changes as each node subscribes and un-subscribes to specific interest sets.

In this architecture, each router must maintain the interest set of each node to which it is connected, including other routers. A router's interest set is the union of all connected nodes. A router then uses the interest state associated with data it receives to determine how to forward the data. For a given topology, communication is minimized such that each client node receives exactly the data in which it is interested.

The initial router implemented in JESPP was a tree router. Each router has multiple clients but only one parent. There is one router that is the top of the tree. A second topology has subsequently been implemented that the authors refer to as a mesh router, which had shown scalability across 1900 nodes in SF Express (Brunett 1998). Instead of a single router at the top of a tree, there is a mesh of routers with "all-to-all" communication. Each simulator is a client of one of the mesh routers. Like the tree router, the primary task of the mesh router is to maintain the interest state of all clients, forwarding only data that is of interest to each client. Further hybrid topologies should be possible with little or no code modification.

The ultimate goal is to establish the capacity of a simulation to scale easily as the number of processors is increased by several orders of magnitude. Comprehensive testing and measurement is required to document the performance of various topologies and router implementations. This testing identifies performance bottlenecks and suggests alternative implementations to be tested. Multiple simulation scenarios are required to construct guidelines for assigning simulators, routers and topologies to multiple SPPs. The simulations and entities had to be designed such that nodes subscribe mainly to a local subset of information.

JFCOM's Experimentation Directorate's recent Joint Urban Operations (JUO) experiments have demonstrated the viability of Forces Modeling and Simulation in a distributed environment. JSAF and the RTI-s communications system, provides the ability to run distributed simulations with sites located from Norfolk, Virginia to Maui, Hawai`i. Interest-aware routers are essential for communications in these distributed environments. The current RTI-s framework provides such routers connected in a straightforward tree topology, which is successful for small to medium sized simulations, but faces a number of significant limitations for large simulations over wide area networks. In particular, traffic is forced through a single site, drastically increasing distances messages must travel to sites not near the top of the tree. Aggregate bandwidth is limited to the bandwidth of the site hosting the top router, and failures in the upper levels of the router tree can result in widespread communications losses throughout the system.

To resolve these issues, this work extends the RTI-s software router infrastructure to accommodate more sophisticated, general router topologies, including both the existing tree framework and a new generalization of the fully connected mesh topologies used in SF Express. The new software router objects incorporate the scalable features of the SF Express design, while optionally using low-level RTI-s objects to perform actual site-to-site communications. General communications protocols are incorporated in the Mesh Router architecture in such a way that it has no impact on the application software. The (substantial) limitations of the original mesh router formalism have been eliminated, allowing fully dynamic operations. The mesh topology capabilities allow aggregate bandwidth and site-to-site latencies to match actual network performance.

1.2 Large Scale Forces Modeling and Simulation

Recent experiments within the Joint Forces Command's Experimentation Directorate, J9, demonstrate the feasibility of forces modeling and simulation applications in a large field of play with fine-grained resolution. As mentioned previously, simulating such battle spaces requires large computational resources, often distributed across multiple sites. The ongoing Joint Urban Operations (JUO) experiment utilizes the JSAF application suite and the RTI-s Run Time Infrastructure to scale to over 300 federates distributed across the United States (Ceranowicz 2002). The JUO exercise has shown the scalability of the JSAF/RTI-s infrastructure and of interest-based, router-managed communication. At the same time, the simulation has highlighted a need for improvements in the communication architecture.

The current JUO network topology is a tree of software routers (see Figure 2 for wide area network diagram). The hub and spoke network model introduced by this tree infrastructure increases latency between distributed sites and exposes the entire network to a single point of failure. The tree topology also poses a scalability limitation within the distributed sites. It is the authors' belief that an improved routing infrastructure is required for the continued success of large-scale entity level simulations, particularly as entity counts as well as complexity and fidelity increase.

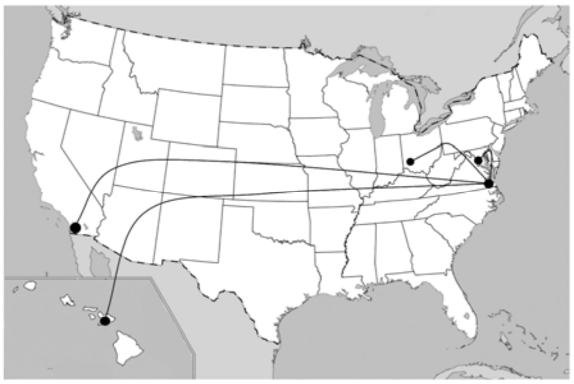

Figure 2: Software Routing Topology for the JUO Exercise

1.3 Scalable Parallel Processors

The JUO exercise requires a computational ability unavailable using traditional groups of workstations. SPPs provide the required computational power, with modest increase in development and execution effort (Lucas 2003). Common SPPs include the IBM SP, SGI Origin, Cray T3E, and the "Beowulf" Linux clusters. Traditionally, SPPs provide services not available in a group of workstations: high speed networks, massive disk arrays shared across the entire resource, and large per-CPU physical memory. In addition, SPPs generally have uniform environments across the entire machine and tools for scheduling, management and scalable interactive control (starting processes across 100 nodes should take the same amount of time as it does across 10).

Linux clusters have recently become suitable platforms for the high performance computing community and are, therefore, readily available at Department of Defense HPCMP Centers. These clusters are ideal platforms for use in the JUO exercise because of their close heritage to the Linux workstations used in the interactive test bays. Although there is additional software to tie the cluster into one SPP, the basic libraries, compiler, and kernel are often the same on a cluster as on a workstation.

1.4 RTI-s

RTI-s provides the HLA Run Time Infrastructure (RTI) for the JUO federation. RTI-s was originally designed in hopes it would overcome the scalability and performance limitations found in RTI implementations at the time and even greater limitations found in the Distributed Interactive Simulation (DIS, IEEE 1278). RTI-s is arguably not a fully compliant HLA/RTI implementation, a matter of less and less consequence these days. Specifically, it does not implement timestamp ordered *receives*, ownership transfer, and Management Object Model (MOM) interactions. In addition, federates discover new objects at first update, rather than at creation time. The JSAF applications are receive-ordered by design and are optimized to respond best to delayed object discovery, so these limitations are not constraining in the existing environment.

Point-to-point modalities in RTI-s use distinctly separate routing processes for communication. The routers provide data distribution and interest management for the federation, which would be too heavy for a simulator to handle. Presently, the tree topology (Figure 3) is used for connecting routers. A tree presents a simple structure for preventing message loops, as there are no potential loops in the system.

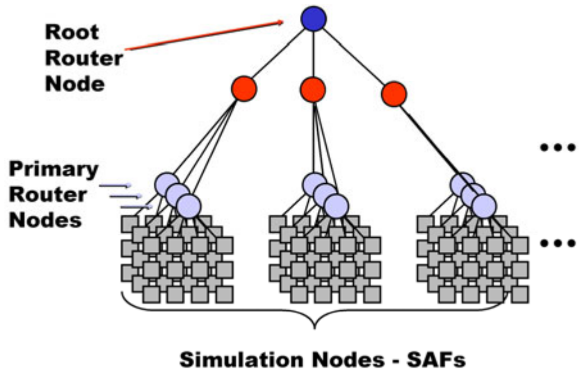

Figure 3: Tree Topology Used by RTI-s for Point-to-Point Message Traffic

The Synthetic Forces Express (SF Express) (Brunett & Gottschalk 1997) project first demonstrated the suitability of both the SPP and mesh router concepts for discrete entity modeling. The SF Express project extended the ModSAF simulation engine (Calder 1993), focusing on the communication protocols to extend scalability.

At the SC'96 conference in November 1996, the SF Express team achieved a 10,000-vehicle simulation using a single 1,024-node Intel Paragon machine. Message routing within the SPP used the Message Passing Interface (MPI) (MPI Forum 1993). Later work allowed the code to run on multiple SPP installations across a variety of networks by introducing gateways between SPPs. The gateway routers were connected using UDP. With these improvements, the project was then able to field a simulation of 50,000 vehicles using 1,904 processors over six SPPs.

The structure of the SF Express router network is shown in Figure 4 These routers distribute (PopUp) and collect (PullDown) messages from client simulators outside the Primary's client set. The SF Express architecture scales to increased problem size by replicating the basic triad and adding full up/down communication links among the triads, as shown in Figure 4. This architecture is the progenitor of the JESPP architecture.

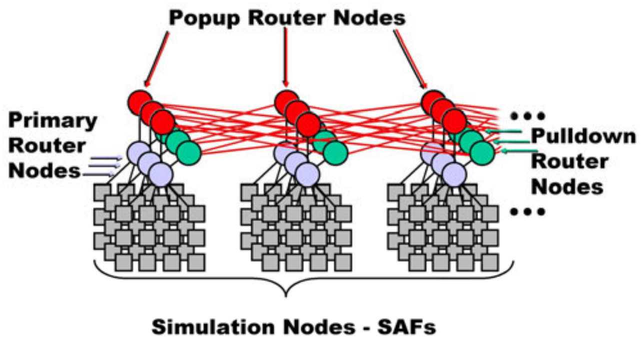

Figure 4: Basic Building Blocks of the Scalable Design Used in the SF Express Routing Network

As a final *magnum opus* accomplishment, the Caltech/ISI/JPL team was able to run a 108,000-entity simulation in March of 1998. These simulated entities were full ModSAF Operational vehicles: tanks, trucks, Bradleys, HumVees, etc. This was demonstrated at a DARPA meeting and was stable enough to be scheduled to be run when it was needed. It used 13 different heterogeneous parallel computers, located from Maryland to Maui, spreading across six time zones.

While the SF Express project was dramatically successful, it had no life beyond a number of 50K-100K entity simulation demonstrations. One issue was the need of one live operator to effectively control each 20 tanks. While the tanks behaviors were correct within local operational areas, they did not have sufficient behavioral sophistication to plan and execute long-range missions. Many of these issues were anticipated, but for a number of reasons they were not seen as invalidating the project, and they did not detract from the proof of concept offered by the success in demonstrating scalability.

This approach seems to have been validated, as new missions have now mandated the provision of several million civilian entities, albeit of limited-functionality. They do not require either sophisticated behaviors or operator control. These are the so-called "clutter" entities that provide civilian entities, such as pedestrians, cars, motorbikes, *etc.* in the simulated environment.

2 DESIGNING FOR SCALABILITY

As previously mentioned, the JSAF/RTI-s application suite currently scales to over 300 federates and over a million entities (including simple clutter). However, current routing topologies limit the scalability of the overall system. In order for an interest-based communication infrastructure to scale, three conditions must hold over an arbitrary interval of simulation time:

- A client must generate a bounded number of messages
- A client must receive a bounded number of messages.
- Given the previous two points, the communication through any given router must also be bounded.

However, any given router must also be bounded, *i.e.* all-to-all communications are not conducive to scalability. An interest management system and careful federate design achieve bounded client communication. Bounded router communication is a function of network design and can be achieved using a mesh topology.

2.1 Interest Management

The aggregate amount of data produced by the JUO federation is greater than any one federate is capable of processing. An interest management system is used to limit the amount of data a federate must process (Rak 1997). The federate declares which information it is interested in ("e.g., red force tanks in position cell X") and the RTI is responsible for ensuring only this subscribed information is received by the federate.

When used in a multicast environment, RTI-s utilizes the concept of multicast channels for filtering, with interest states having associated channels. The receiver filters the message at the kernel level, so the application never sees messages for interest states in which it is not interested. Due to the limited number of available multicast channels, the number of interest states is limited (increasing the amount of traffic associated with each interest state).

The ISI/Caltech team developed a more effective design. When running in point-to-point mode, interest management is "send-side squelched." Software routers maintain interest state vectors for each connection and only send messages to clients that have expressed interest in a message type. Because interest states are not tied to hardware and operating system limitations, the number of available interest states is comparatively unlimited. This is an enormous improvement over multicast IP. It was also one of the innovations of SF Express.

2.1.1 Routing Scalability

The scalability of the basic Mesh Router network is easily argued as follows. It is first necessary to assume that the underlying simulation problem itself has a scalable solution. This means a bounded message rate on the Primary $\Rightarrow$ PopUp and PullDown $\Rightarrow$Primary links within a basic triad, and bounded Up $\Rightarrow$ Down message rates within the interconnection links of the full network. The impediments to complete scalability of the mesh architecture have to do with interest declarations among the upper router layers. Each PullDown must announce its interest to every PopUp.

In principle, these interest broadcasts could be made scalable through an additional network of communication nodes. In practice, however, these interest updates were not frequent enough to cause any difficulties in SF Express or the JFCOM simulations with as many as thirty triads in the full mesh. An experiment with a similar mesh router setup using the current infrastructure shows similar results.

2.1.2 Routing Flexibility

The scalability issues with the tree router topology of RTI-s have been discussed previously. Tree topologies also map poorly onto physical wide-area networks. Figure 2, above, shows the route taken for any message crossing multiple sites in the JUO exercise. The path taken for a message to go from Maui to San Diego is sub-optimal: the data must first travel to Norfolk, then back to the west coast. This extra transmission time increases the latency of the system, which lowers overall performance. Since wide-area links often have less bandwidth available than local area networks, such routing also places a burden on the Virginia network infrastructure, which must have bandwidth available for both the incoming and outgoing message in the authors' Maui to San Diego example.

The mesh routing infrastructure provides a better utilization of physical networks by sending directly from one source to destination router. The network infrastructure is free to route messages in the most efficient way available. Figure 5 shows one possible routing topology for the JUO exercises, using mesh routers to minimize the distance messages must travel.

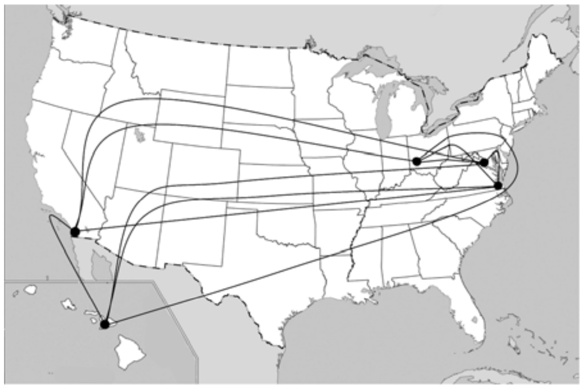

Figure 5: Advanced Routing Topology for JUO Exercises

In an ideal world, the entire federation would use one fully connected mesh for message routing. The actual routing of messages would be left to the physical network infrastructure. This Internet technology has over 30 years experience in optimizing data. However, such a configuration is often not feasible due to performance or protocol availability. Local area communication is usually over TCP, pushing error detection from RTI-s to the network stack. Over wide area networks, however, TCP suffers bandwidth degradation proportional to latency, so UDP is used for these connections. Some SPPs provide neither TCP nor UDP on computer nodes, instead providing MPI over a high-speed network, or provide public access only on a small subset of the machine.

The mesh router provides the ability to design a flexible network topology that meets the constraints of the network infrastructure while providing the ability to design a scalable system. The mesh router's topology is constructed by combining two building blocks: a tree (Figure 6, left) and a fully connected mesh (Figure 6, right). The two building blocks can be combined to form meshes of meshes, trees of meshes, meshes of trees, etc. The process can be repeated as often as required to build a suitable topology, but the mesh is scalable.

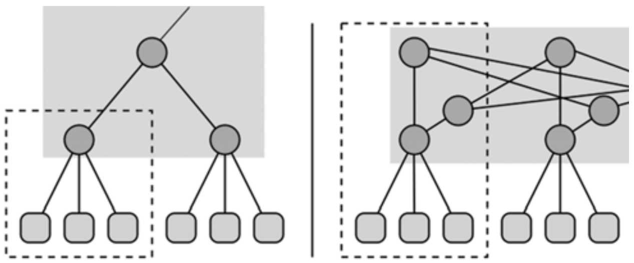

Figure 6: The Basic Building Blocks for a Mesh Router Topology: Tree (left) and Mesh (right)

2.2 Mesh Router Architecture

The mesh routers developed for RTI-s adopted many of the design decisions made in the SF Express project. The router triad concept is perhaps the most obvious of the design decisions from SF Express, providing an elegant method of avoiding "message looping" in the mesh, while allowing an arbitrary number of routing decisions to be made when transferring messages. However, significant design changes have produced a radically more advanced and flexible infrastructure.

2.2.1 Comparing Tree and Mesh Routers

In an analog of the decision to use HPC rather than just staying with workstation technology, one can easily decide that the straightforward Tree Router design is initially adequate, but eventually, one is confronted with the need to optimize. When the users demand new capabilities, more entities, or behavioral veracities, the potential benefits of the Mesh Router are not only desirable, they quickly become necessary.

2.2.2 Flow Control

A tight flow control with Request to Send / Clear to Send (RTS/CTS) behavior was used in the SF Express design.

SF Express used the mesh routers within SPPs, where latencies were extremely low and available bandwidth greatly exceeded expected message transfer rates. The overhead of sending the RTS and CTS messages would not negatively impact the performance or scalability of the system. The communication medium of choice (MPI) requires pre-posted receive buffers of a known size, requiring a RTS/CTS protocol for sending large messages. However, recent trends have shown CPU power improvements far outpacing network latency and bandwidth improvements. On modern networks, a RTS/CTS protocol poses a significant performance burden. Therefore, the Mesh Router architecture has an eager send protocol with messages dropped by priority when queues overflow.

2.2.3 Application-Independent "Message" and "Interest" Objects

The Mesh Router software is object-oriented (C++), with a limited number of standard interfaces to "user message" and "interest" base classes. For present purposes, the implications of this factorization are:

- The Mesh Router system is designed to be compatible with ongoing changes and evolution within the RTI-s system, requiring little more than "re-compile and re-link".
- The Mesh Router system can support applications other than SAF/RTI, given appropriate different instances of the message and interest objects.

2.2.4 Factorized Communications Primitives

The Mesh Router object design relies on a very careful isolation/factorization of the underlying message exchange protocol from the rest of the JSAF software. The essential object design is conceptually indicated in Figure 7 and has three layers:

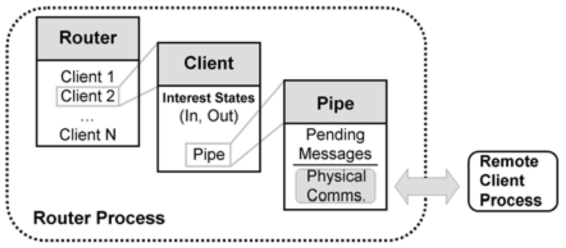

Figure 7: Schematic Design of the Mesh Router Application

2.3 Performance Testing Router Architectures

In a continuing effort to further quantify the relative utility of advanced router designs, new performance runs were accomplished in the first quarter of 2005. ASC-MSRC's Linux Cluster, Glenn, is basically the same configuration as Koa at MHPCC. In the case of the cluster at Wright-Paterson, there are 60GB hard disks on each local node, a configuration subsequently installed on MHPCC's Koa.

The implemented simulation utilized for the Glenn performance studies again used timed message exchanges between pairs of simple federates. One processor within each pair initiates a sequence of fifty fixed-size message exchanges with its partner, adding the times for each "there and back" message exchange. The process is repeated for a number of different message sizes. The primary output for each master-slave pair is simply a list of average exchange times versus message size.

A baseline set of measurements were taken using the Tree Router architecture. Runs were conducted at differing message sizes and communications were monitored between nodes that varied the number of Tree Routers through which they had to pass (hops) from one node to the other. Similar runs were conducted for the Mesh Routers, using the same terminology to describe the relative separation of the nodes on the mesh, even thought the actual hops required became a misnomer, as the Mesh Routers always pass through the same number of routers (Gottschalk 2005)

Figure 8 presents the finding that MeshRouter 2-Hop and 3-Hop message delivery times are typically 2-4 times faster than the tree router in the 1Kbyte-10Kbyte message range, typical in JSAF. The bi-modal timing distribution indicates significant contention, as the individual messages are all pushed through a limited number of high-level routers. (Put differently, 20% of the communications pairs are left waiting while the first arrivals get out of the way). MeshRouter performance measures are remarkably insensitive to the Master/Slave separations. Note that with 0 hops, there is no significant difference between the architectures. As the size of the simulation grows, either due to span or complexity of the scenario, the advantage of the mesh routers becomes more evident. These differences persist up beyond 10,000 Byte messages.

TreeRouter performance degrades substantially as the underlying physical message path increases, the MeshRouter scales with point-to-point performance that is largely insensitive to the overall problem size. TreeRouter performance degrades substantially as the underlying physical message path increases, the MeshRouter scales with point-to-point performance that is largely insensitive to the overall problem size.

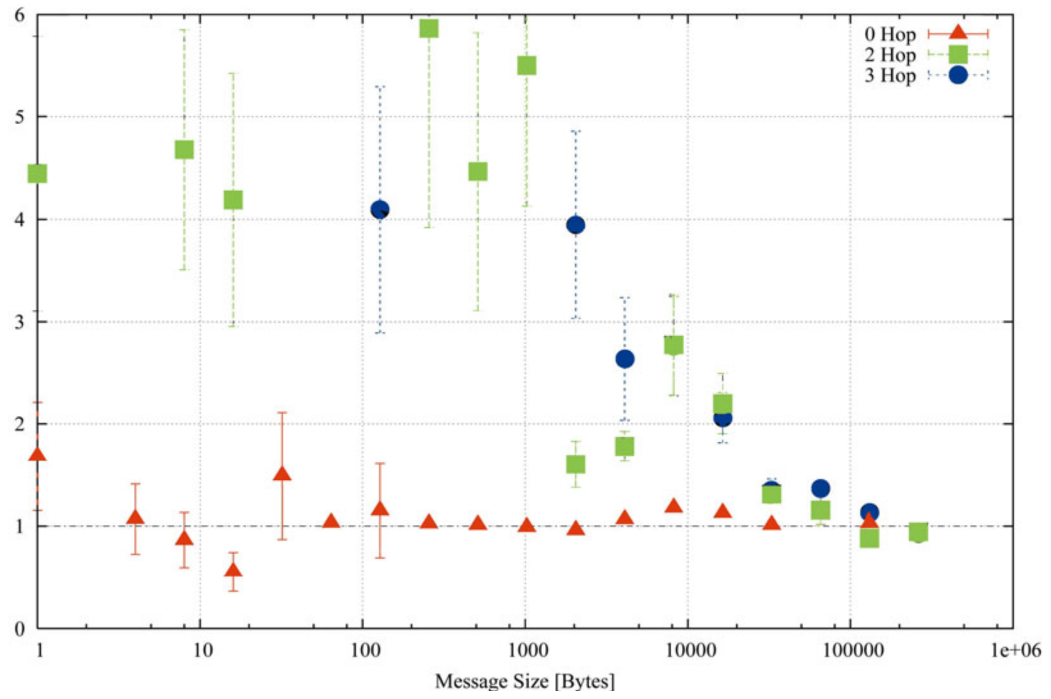

Figure 8: Relative Enhanced Throughput for MeshRouters versus TreeRouters

3 DATA COLLECTION AND MANAGEMENT

3.1 Baseline Plan for Processing Collected Data

The simulation would be of little and transitory use if the data collected could not be probed and studied. The initial design created by the JFCOM team used a centralized logging approach. In this design, a single logging process subscribes, through RTI interest management, to all messages published by the simulators. This all-to-one communication scheme performed reasonably on local area networks with a small number of simulators. But this approach does not scale to hundreds of simulators running on multiple clusters. In such computing environments, the centralized logger becomes the bottleneck. The physical network and routers cannot keep pace with the volume of messages. They are forced to drop simulation messages. Even if the network was able to deliver the messages to the centralized logger, the logger will not be able to store the messages and to process them for analysis in a timely manner.

The initial implementation of the design was limited by the use of the Access2000® database, which had an internal limit of 2GB. This forced the collected data to be stored in separate data segments. After a simulation run was completed, the collected database segments were processed into the single MySQL database in the following sequence: First, a database schema was applied to the MySQL database that would represent the "rollup" of all of the data collected. Second, the Access2000 database segments were put into sequential order by their unique file names and internal file creation timestamps. Third, each segment was filtered to build a list of relevant tables from which data is to be extracted. Fourth, the first Access2000 database was inserted directly into the MySQL database. Fifth, subsequent Access2000 databases were processed to ignore overlapping data by examining the difference in timestamps between where the previous segment ended and the next segment began. Also, internal record timestamps were adjusted with an offset so that individual record timestamps represented time since beginning of simulation run and not just for the individual database segment. Finally, summarization information was extracted and stored into new tables to facilitate speed in reviewing common information. Those reports that were processing-intensive will be generated and saved for post-event review.

3.2 Post-Event Data Processing

Once the simulation data had been processed and inserted into the MySQL database, the MOP/MOE (Measure of Performance/Measure of Effectiveness) tools were applied to the completed database to provide predefined statistics for the event period. In conjunction with these predefined reports, additional reports and queries can be rapidly created based on additional feedback and desires of the analyst. A sample of predefined reports available for the end user included: Killer/Victim scoreboard, Entity Lifecycle and Lifecycle Details, and "String" analysis charts. Other MOP/MOE components were developed and included with the toolkit based on input from the data collection and analysis plan designed by the joint experimentation users and analysts.

3.3 Implementation of Enhancements

The ISI/IDA data team made changes to the baseline plan for data collection and analysis as new challenges arose along the way. The key challenge is to provide the capability to log all simulation messages without adversely affecting and interfering with the performance of the JSAF simulation. The approach taken is to implement a distributed logger that minimizes network communication overhead by logging the data at the data generation source, and by selectively propagating that data, based on need.

Instead of one centralized logger, a local logger was created for each simulator. Conceptually, this local logger intercepts messages emitted by the local simulator to the RTI communication layer. The intercepted messages are stored locally on the compute node. No network logger traffic is generated during the logging process. As long as each local logger can keep up with the local simulator, this approach provides seemingly perfect scalability with respect to the number of simulators. As more simulators are added more local loggers are added. There seem to be no centralized bottlenecks in this approach.

3.3.1 Interceptor/Logger

The Interceptor/Logger application is a process that resides on individual simulation nodes within the federation. The determining factor on where to utilize the mechanism is determined by which federates are publishing information needed for data collection. The interceptor/logger, utilizing functionality in the RTI Application Programming Interface, inserts "hooks" into the published data streams by the RTI and then splits off two child processes, one process that writes and compresses the intercepted data into binary "log" files. A second process is one that decodes the data stream and inserts the decoded data into an embedded database application, initially SQLite, now MySQL. A separate daemon process called "sqlited" handles incoming socket-based connection attempts to query information that has been stored in the local database. Figure 9 is a diagram of the process.

Because of the methodology of running interceptor/loggers on each simulator with data of interest, a separate mechanism was needed to retrieve information stored at each simulator location. A separate application process called "Aggregator" was developed that would handle the intercommunication between simulators logging data. The Aggregator is configured in a tree-like fashion, with a "Root Aggregator" at the head of the tree and "Child Aggregators" in branches from the root. The various branches reach out to the individual leaf instances of "sqlited" on each simulator. The interface to the Root Aggregator takes

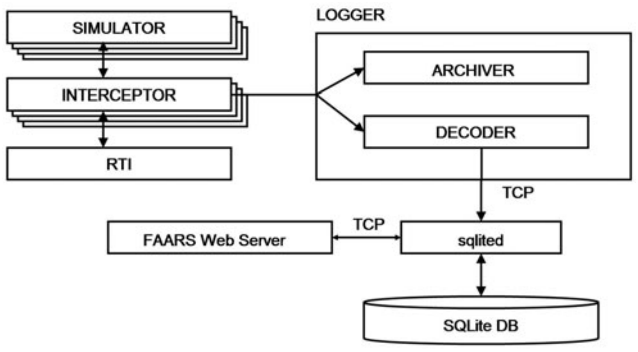

Figure 9: Interceptor/Logger Process

a Structured Query Language-formatted query and passes the query on to each of the branch Child Aggregators until the query finally reaches the individual instances of "sqlited". As each instance of "sqlited" responds with the requested data for the query, the Child Aggregators assemble the returned information in order of response and forward the data on to the Root Aggregator, which then assembles the complete, returned information and forwards it on to the original requestor. The Aggregator model works on Transmission Control Protocol (TCP) socket-based connections between the Root Aggregator and subsequent Children Aggregators.

3.3.2 Near Real Time Retrieval Of Data

With the utilization of the ISI interceptor/logger, the possibility of retrieving simulation information in a "near real time" manner became a reality. Typically, data collection efforts have had to wait until hours after collected logger files have been processed before any specific event information could be derived. This is a vast improvement in functionality and provides a wide range of uses that are still being realized as we move forward in the software development effort.

The Near Real Time data retrieval effort is based around the ability to query the ISI interceptor/logger application, retrieve the logged information from each node and store the retrieved information into a local Relational Database Management System (RDBMS). The retrieved information is then used by the FAARS Near Real Time web server interface to allow users of the system to view various reports, charts and graphs based on the available information.

The process of retrieving intercepted information from each of the active ISI interceptor/loggers is handled by a series of BASH shell scripts on the FAARS web server. Each BASH shell script is targeted towards retrieving specific information, such as entity object states, and is used to process the retrieved information into the local RDBMS (aka cache). The data retrieval process is based on three steps. The first step is to send the request for information to the Root Aggregator. The methodology used by the retrieval process is based on making a TCP socket-based connection to the Root Aggregator and sending an SQL-formatted query. The second step is to wait for a response and process/validate the retrieved data and write this data to a temporary file. The response from the Root Aggregator is a stream of plain ASCII text, which is tab-delimited for fields and is carriage return delimited for individual records. This information is then written to a temporary file in "tab-delimited/carriage return delimited" format. The third and final step is to load the temporary file's data into the local cache.

The FAARS web server RDBMS cache now uses MySQL (v4.1.1.) as the database engine. The database schema for the cache is based primarily on the schema used by the ISI interceptor/logger. This helps in facilitating compatibility with the information that is being utilized in near real time and data being reviewed post event. The main difference between near real time and post event processing is the different indexing schemas utilized on the local cache. The indices applied to the local cache database have been specifically tuned to support the types of queries that the FAARS web server uses for data displays.

Because of the nature of distributed logging of the data that is currently implemented, it is now necessary to develop means to:

- retrieve all of the saved binary data logs on each simulator, with the ISI interceptor/logger
- prepare and decode the binary data files
- insert the decoded data into a consolidated database (the complete accumulation of data for a that event.)

A process called "Data Staging" has been developed that accomplishes these tasks in an organized, efficient manner, making the best usage of available bandwidth, processing cycles and disk space. The Data Staging process begins with retrieving the binary log files at the end of each day's simulation run from each simulator logging data. The data is moved and stored on the local storage point in a hierarchical format based on the event name, day of the event and the simulator where the log file was retrieved. Once the data has been moved, Perl-based scripts are run against the individual binary log file to decode and format the binary data into plain-text, comma-separated value (CSV) flat files. The translation of the data and the creation of the storage database schema are based on utilizing definitions found in the Federation Object Model (FOM) and Federation Execution Document (FED) for the federation in use. Each CSV-formatted file represents a section of data to be inserted into the consolidated database for the event. A final Perl-based script takes the CSV-format files and inserts the decoded data into the appropriate table within the consolidated database.

4 ACCOMPLISHMENTS AND FUTURE DIRECTIONS

4.1 Accomplishments

In December of 2002, the JESPP team ran a successful prototype event using a partition of the USC Linux cluster, consisting of some 240 servers, with 2 GHz Xeons, 1 GByte of RAM and both GigE and Myrinet mesh communications. The scientists at ISI in California and the operators at JFCOM in Virginia jointly shared control. More than 1,000,000 civilian entities were successfully simulated. They showed appropriate behavior and were stable, even when scanned by the SLAMEM program, emulating two GlobalHawk platforms. To ensure usability and operational validity, about 1,100 warfighting entities were also simulated and controlled in a manner consistent with normal J9 experimentation. Stability and appropriate response to control commands were evident throughout. Several runs were conducted over the course of a week and performance was characterized.

Following the December event, it was decided to show the utility of the DoD's SPP assets by using two Linux clusters, at two High Performance Computing Modernization Program sites, MHPCC and ASC-MSRC. The HPCMP then established a new Distributed Center (DC) to provide computing assets for this work. The senior staffers of the HPCMP were most helpful in discussions relating to the appropriate hardware for the task at hand. As an illuminating example, they raised the issue as to the potential benefits of installing 64 bit AMD or Intel processors, but were very sensitive to the ultimate customer's desire that the leap to high-end computing and parallel processing on Linux clusters remain as closely compatible with the existing Linux workstation configurations. The system was installed in the spring of 2004 and became operational within weeks.

Issues including the location of the Linux cluster, government security, networking bandwidth and latency, and cluster capabilities were surfaced and resolved. The desire for both redundancy and the desirability of further proving distributed high performance computing militated in favor of splitting the 256 nodes (two processors, 4GB RAM per node) into two machines, one at the MHPCC in Hawai'i and one at the ASC MSRC in Ohio. Both of these goals have been met and the original analysis has been proven prudent many times, when one site was able to carry the entire load while the other was faced with some type of outage, administrative encumbrance (*e.g.* Center power system maintenance) or network isolation.

Networking turned out to be a small factor and evidenced itself in unexpected ways. The underlying simulation program, one of the SAF family, tolerates latencies quite well. Even latencies up to 500 milliseconds (0.5 seconds) are tolerable. However, the speed-of-light-latency alone from Virginia to Maui is on the order of 100 milliseconds, and that was disorienting for the operators, not in the performance of the simulation, but in graphical user interface (GUI) issues like the delay in a drawn circle responding to mouse/cursor movements.

4.2 Open Issues for Future Work

There is much to be done in terms of instrumenting and analyzing the existing system, contrasting performance with that from communications options within the current RTI-s baseline. The more interesting studies here will involve comparisons of new qualitative features of the underlying simulations. An example is the difference between "reduced capability" and "self-aware" clutter (*i.e.*, do clutter objects interact). Many of the more interesting near-term development paths can be characterized in terms of "special purpose gateways".

5 CONCLUSION

While the techniques proposed in this paper may not be a universal panacea, the authors maintain those techniques will increase the opportunity for the defense analysts to discover new and dangerous threats and work out the best way to defend against the destruction from those threats.

The mesh router infrastructure presents a scalable routing infrastructure for both local and wide area communication. They are capable of being organized into a number of topologies and should be easily extensible. For wide area networks, the flexible routing topologies allow communication over all available network links, without the hub and spoke problem of the tree routers. Within a local area network, the mesh routers provide a scalable communication architecture capable of supporting hundreds of federates.

This was not merely a translation of existing communications procedures. It was the first of a number of steps to achieve the qualitatively new capabilities that follow from the scalable communications of the basic architecture. It will also provide capabilities of the "intelligent gateways" for WANs, supportable within this architecture.

ACKNOWLEDGMENTS

The authors would like to thank the Joint Experimentation Group at JFCOM and they acknowledge the support and encouragement of Major General Woods, Anthony Cerri, and Jim Blank. The authors especially would like to acknowledge the direction, support and counsel of Rae Dehncke, program manger and guiding light for this project. They would like to recognize the excellent technical support of the ISI's Computational Sciences Division, especially John Tran and the guidance and assistance from ISI's Scalable Systems Division and the JSAF team: Dr. Andy Ceranowicz, Mark Torpey, Bill Hellfinstine and many others. This material is based on research sponsored by the Air Force Research Laboratory under agreement numbers F30602-02-C-0213 and FA8750-05-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes, notwithstanding any copyright notation thereon.

REFERENCES

Barrnett, T.P.M. 2004. *The Pentagon's New Map: War and Peace in the Twenty-First Centur*y, New York, New York, Putnam Company.

Ben-Ari, E. 1998. Mastering soldiers: conflict, emotions and the enemy in an Israeli military unit, *New Directions in Anthropology*, V. 10. Oxford: Berghahn Books.

Brunett, S., D. M. Davis, T. D. Gottschalk, & P. Messina. 1998. Implementing distributed synthetic forces Simulations in metacomputing environments, *Seventh Heterogeneous Computing Workshop*, Orlando, Florida.

Brunett, S., and T. D. Gottschalk 1997, Scalable ModSAF simulations with more than 50,000 vehicles using multiple scalable parallel processors, Technical Report CACR-156, 1997, Caltech, presented at *Spring Simulation Interoperability Workshop*, Orlando, Florida.

Calder, R. B., J. E. Smith, A. J. Courtemanche, J. M. F. Mar, and A. Z. Ceranowicz. 1993. ModSAF behavior simulation and control. In *Proceedings of the Third Conference on Computer Generated Forces and Behavioral Representation. Orlando, Florida: Institute for Simulation and Training*, Univ. of Central Florida.

Cebrowski, A.K., and J.J. Garstka, 1998. Network centric warfare: its origin and future, *Naval Institute Proceedings*, 124/1, 28-35, Annapolis, Maryland.

Ceranowicz, A. Z., M. Torpey, W. Hellfinstine, J. Evans, and J. Hines. 2002. Reflections on building the joint experimental federation, *Proceedings of the 2002 I/ITSEC Conference*, Orlando, Florida.

Ceranowicz, A. and M. Torpey. 2004. Modeling human behaviors in an urban setting, *Proceedings of the 2004 I/ITSEC Conference,* Orlando, Florida.

Dahmann, J., J. Olszewski, R. Briggs, and R. Weatherly. 1997. High Level Architecture HLA performance framework, *Fall 1997 Simulation Interoperability Workshop*, Orlando, FL, 1997.

Davis, D. M., G. D. Baer, and T.D. Gottschalk. 2004. 21st Century simulation: exploiting high performance parallel computing and advanced data analysis, *Proceedings of the 2004 I/ITSEC Conference*, Orlando, Florida

Defense Modeling and Simulation Office. 1998. *High Level Architecture Interface Specification*, v1.3, 1998.

Foster, I. and C. Kesselman 1997. Globus: a metacomputing infra-structure toolkit, *Intl Journal Supercomputer Applications*, 112: 115 –128

Fujimoto, R. and P. Hoare. 1998. HLA RTI performance in high speed LAN environments, in the *Fall Simulation Interoperability Worksho*p, Orlando, Florida.

Garlan, D. and M. Shaw. 1993, An *Introduction to Software Architecture: Advances in Software Engineering and Knowledge Engineering*, volume I. New York, New York, World Scientific Publishing.

Gottschalk, T. D, P. Amburn, and D. M. Davis. (Forthcoming, 2005), "Advanced message routing for scalable distributed simulations," *The Journal of Defense Modeling and Simulation*, San Diego, California, Publication pending

Hill, R. W., J. Gratch, and P.S. Rosenbloom. 2000. Flexible group behavior; virtual commanders for synthetic battlespaces. *Proceedings of the Fourth International Conference on Autonomous Agents,* Barcelona, Spain.

Kaufman, W. and L. Smarr. 1993. *Supercomputing and the Transformation of Science,* New York, Scientific American Library

Keahey, K. and D. Gannon. 1997. PARDIS: A parallel approach to CORBA, *Proceedings. The Sixth IEEE International Symposium on High Performance Distributed Computing*, pp: 31-39, Portland, Oregon

Lucas, R. F. & Davis, D. M. 2003,. Joint Experimentation on Scalable Parallel Processors. *In Interservice/Industry Training, Simulation, and Education Conference*, Orlando, Florida.

Messina, P. C., S. Brunett, D. M. Davis, and T.D. Gottschalk. 1997. Distributed interactive simulation for synthetic forces, In J. Antonio, Chair, *Mapping and* Scheduling Systems, *International Parallel Processing Symposium*, Geneva, Switzerland.

MPI Forum 1993. MPI: A message passing interface. In *Proceedings of 1993 Supercomputing Conference*, Portland, Washington.

Rak, S., M. Salisbury, and R. MacDonald. 1997. HLA/RTI data distribution management in the Synthetic Theater of War, *Proceedings of the Fall 1997 DIS Workshop on Simulation Standards*, Orlando, Florida

Sanne, J. 1999. *Creating Safety in Air Traffic Control*. Unpublished doctoral dissertation, Institute of Tema Research, Linköping University, S-581 83 Linköping, Sweden.

van Lent, M., and K. Laird, 1998. Learning by observation in a complex domain. *Proceedings of the Knowledge Acquisition Workshop*, Banff, Canada.

AUTHOR BIOGRAPHIES

GENE WAGENBRETH is a Systems Analyst for Parallel Processing at the Information Sciences Institute at the University of Southern California, doing research in the Computational Sciences Division. He specializes in tools for distributed and shared memory parallelization of Fortran programs and has been active in benchmarking, optimization and porting of software for private industry and government labs. genew@isi.edu.

KE-THIA YAO is a research scientist in the Distributed Scalable Systems Division of the University of Southern California Information Sciences Institute. Currently, he is working on the JESPP project, which has the goal of supporting very large-scale distributed military simulation involving millions of entities. Within the JESPP project he is developing a suite of monitoring/logging/analysis tools to help users better understand the properties of large-scale simulations. kyao@isi.edu.

DAN M. DAVIS is the Director, JESPP Project, Information Sciences Institute (ISI), University of Southern California, and has been active for more than a decade in large-scale distributed simulations for the DoD. While he was the Assistant Director of the Center for Advanced Computing Research at the Caltech, he managed Synthetic Forces Express, a multi-year simulation project and pursued other simulation interests. ddavis@isi.edu.

ROBERT F. LUCAS is the Director of the Computational Sciences Division of ISI at the University of Southern California. There he manages research in computer architecture, VLSI, compilers and other software tools. He has been the principal investigator on the JESPP project since its inception in 2002. rflucas@isi.edu.

THOMAS D. GOTTSCHALK is a Lecturer in Physics at the California Institute of Technology and a Member of the Professional Staff, Center for Advanced Computing Research there at Caltech. His research has been in the field of the use of parallel computers to simulate various physical phenomena. tdg@cacr.caltech.edu.