AFRL-IF-RS-TR-2004-273
In-House Report
October 2004

DISTRIBUTED INFORMATION ENTERPRISE MODELING AND SIMULATION (DIEMS)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2004-273 has been reviewed and is approved for publication

APPROVED: /s/

JAMES HANNA
Project Engineer

FOR THE DIRECTOR: /s/

JAMES A. COLLINS, Acting Chief
Information Technology Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	OCTOBER 2004	In-House Report Sep 01 – Mar 04

4. TITLE AND SUBTITLE
DISTRIBUTED INFORMATION ENTERPRISE MODELING AND SIMULATION (DIEMS)

5. FUNDING NUMBERS
C - IN-HOUSE
PE - N/A
PR - 450T
TA - JB
WU - IP

6. AUTHOR(S)
James Hanna and
Robert Hillman

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/IFTC
26 Electronic Parkway
Rome New York 13441-4514

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/IFTC
26 Electronic Parkway
Rome New York 13441-4514

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2004-273

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: James Hanna/IFTC/(315) 330-3473/ James.Hanna@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***
The Air Force is developing a Distributed Information Enterprise Modeling and Simulation (DIEMS) framework under sponsorship of the High Performance Computer Modernization Office Common High Performance Computing Software Support Initiative (HPCMO/CHSSI). The DIEMS framework provides a design analysis environment for deployable distributed information management systems. DIEMS establishes the necessary analysis capability allowing developers to identify and mitigate programmatic risk early within the development cycle to allow successful deployment of the associated systems. The enterprise-modeling framework builds upon the Synchronous Parallel Environment for Emulation and Discrete-Event Simulation (SPEEDES) foundation. This simulation framework will utilize "Challenge Problem" class resources to address more than five million information objects and hundreds of thousands of clients comprising the future information based force structure. The simulation framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance. SPEEDES provides an ideal foundation to support simulation of distributed information systems on a multiprocessor platform. SPEEDES allows the simulation builder to perform optimistic parallel processing on high performance computers, networks of workstations, or combinations of networked computers and HPC platforms.

14. SUBJECT TERMS
Information Enterprise, Enterprise Modeling, Distributed Simulation, High Performance Computing

15. NUMBER OF PAGES
45

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

TABLE OF CONTENTS

LIST OF FIGURES

Distributed Information Enterprise Modeling and Simulation (DIEMS)

The Air Force is developing a Distributed Information Enterprise Modeling and Simulation (DIEMS) framework under sponsorship of the High Performance Computer Modernization Office Common High Performance Computing Software Support Initiative (HPCMO/CHSSI). The DIEMS framework provides a design analysis environment for deployable distributed information management systems. DIEMS establishes the necessary analysis capability allowing developers to identify and mitigate programmatic risk early within the development cycle to allow successful deployment of the associated systems. The enterprise-modeling framework builds upon the Synchronous Parallel Environment for Emulation and Discrete-Event Simulation (SPEEDES) foundation. This simulation framework will utilize "Challenge Problem" class resources to address more than five million information objects and hundreds of thousands of clients comprising the future information based force structure. The simulation framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance. SPEEDES provides an ideal foundation to support simulation of distributed information systems on a multiprocessor platform. SPEEDES allows the simulation builder to perform optimistic parallel processing on high performance computers, networks of workstations, or combinations of networked computers and HPC platforms.

Future DoD campaign strategies are being developed based on the concept of deployable distributed information management systems that are integral parts of the overall deployment footprint [1,2]. This integrated information system will provide assets and individual users with time-critical information required for their function during the crisis or conflict. This new deployable information enterprise will allow DoD forces to maintain information superiority over adversaries and react accordingly. The DoD will invest substantial resources over the next decade on the development of the distributed information enterprise infrastructure.

The Air Force is developing a concept known as the Joint Battlespace Infosphere (JBI) [3] to achieve information superiority. The other Military Services are developing similar concepts, such as the Army's "Digitized Battlefield" [4] and the Navy's "Network-Centric Warfare" [5]. These systems will move the state-of-the-art in information management from the current network-centric warfare designs into information-centric warfare, thus providing raw and fused information to end users at presently unachievable quality and rates. Further, individual Military Services integrated information enterprise architectures must employ system-of-systems concepts to expand their application to include Joint and Coalition Task Force deployment systems as well.

No single information architecture can or will be established to meet these operational requirements. Numerous architectures will be developed and implemented. These architectures are expected to continually morph to reflect the most current technology

trends through their lifetime. The overall requirement is that all implementations interact seamlessly. This three-year effort leverages previous DoD investments in event level simulation by building a generalized information architect simulator on top of SPEEDES [6,7].

Military Service implementations of the enterprise architectures will initially be created working with prototypes and eventually culminate in simulation of fielded implementations. The information enterprises will provide scalability challenges for currently available information management hardware and software systems and design requirements to create more capable systems. Each prototype and implementation will require numerous resources, including physical network implementations, as a basis to support design, development, and operational efforts. Important issues such as computational requirements, storage, information protection and assurance, bandwidth and connectivity will be addressed. Significant effort will be required to assess and evaluate these potential resource requirements. The distributed information enterprise must be simulated based on the operational behaviors of the members in conjunction with the physical constraints of the resources.

This project is part of the HPCMO/CHSSI System-of-Systems portfolio developing scalable software that emphasizes the integration of autonomously operating weapon systems into a dynamically controlled information network or System-of-Systems (SOS). In recent years, the Department of Defense (DoD) has recognized that weapon systems operating autonomously provide a less than optimal solution to our national security problems. Information processing nodes in the network will fuse information from other nodes to provide a relevant battlespace view to friendly participants. Due to the complexity of these systems, development and implementation of this capability will rival any other difficult development performed by DoD.

Testing of SOS will require simulations more complex than any developed to date. The tracking of interaction between hundreds of thousands of players, complex weapons systems, and environmental models while merging physics with information theory can only be achieved through the aggressive application of high performance computing and networking technologies. The complementary applications in this portfolio address virtual system-of-systems characteristics ranging across the multiple domains. These efforts include: (1) Prototyping and Scalability Assessment of Distributed Information Enterprise, (2) Virtual Communication Links, and (3) Virtual Prototyping and Accelerated Testing of Systems. The portfolio efforts will all utilize SPEEDES as the underlying simulation framework and will be interoperable.

An efficient parallel, portable, flexible, scalable, and commercial quality simulation environment is required for the development of complex deployable information systems. The research reported here will develop and implement this capability. The DIEMS simulation framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance on the scale necessary to meet the aggressive requirements of DoD information enterprises.

The DIEMS architecture is intended to provide the framework, mechanism, and semantics to support the modeling and simulation of a wide variety of information enterprise functionality. The design of a generalized simulation engine to support the modeling of a diverse set of distributed information systems poses considerable technical challenges. The modeling framework provides the user with the ability to separate the specification of nodal functionality from physical topology and communication protocols. Further, the framework provides the flexibility necessary to specify a broad set of nodal functionalities and physical topologies. The user may specify the system communication and operational scenarios necessary to drive the simulation and validate the model. Most users will be free from the necessity of writing code by defining model definitions for the information system enterprise in a declarative manner using configuration files. The modeling framework will simulate the specified information enterprise performance at the information level to identify, quantify, and resolve protocols, processes, and core functions as part of the prototyping, development, and scalability assessment of an information enterprise (see Figure 1).

The DIEMS interconnection network defines the bandwidth (or capacity) and latencies (or delay) of the connections between clients. Creating collections of clients with specific processing capabilities and networking characteristics can be achieved through instantiation with static parameters. Multiple network domains are modeled by associating sets of clients with a given network domain and defining bandwidth and latency as a function of domain locality. In this approach, aggregating multiple network domains is achieved simply by partitioning the enterprise by domain association. Again, this can be achieved through instantiation with static parameters using a configuration file.

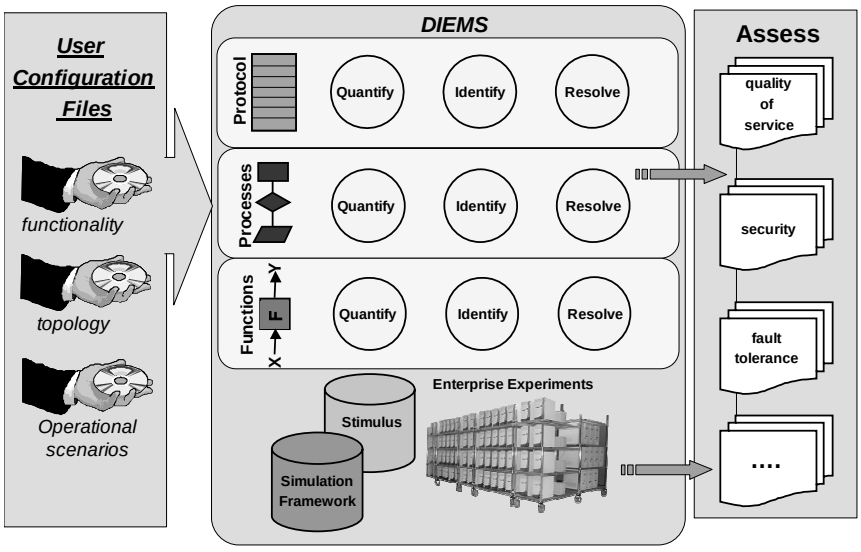

Figure 1. DIEMS Modeling Framework.

Distributed information systems consist of a collection of autonomous clients interconnected via a network. Each client provides some product (or functionality) to the enterprise, and perhaps subscribes to (or consumes) one or more enterprise products. The specification of system topology is rather straightforward. The specification of individual client functionality, however, is not so simple. A given enterprise model may consist of

thousands of clients distributed across tens or even hundreds of network domains. Client functionality cannot be achieved through static parameter instantiation; it requires a more sophisticated mechanism.

Our architectural approach is to provide the underlying mechanisms to allow users to specify the behavior of each client as a process table. This process table will be a local client data structure built during the initialization phase of the simulation and will be based on a configuration file. The processes enumerated in the table are callable methods that define atomic actions. Each process in the table may also consist of pre- and post-conditionals, as well as parameters necessary to fully define the atomic action desired. The order and aggregation of a given set of these processes will define the functionality of the client. Each client will iterate through its process table when it receives an activation event. The process pre-conditional may be used to determine whether a given process is active during the given cycle. The process post-conditional may be used to trigger events or set pre-conditionals of subsequent processes. This approach will allow a broad set of functionality to be specified for clients.

All atomic actions are defined as public C++ class methods. Data types and objects, including conditionals, are defined using C++ container classes that are inherited by the atomic actions class. There is a one-to-one mapping between the elements specified in the configuration file and the methods defined in the atomic actions class. Instantiation of an atomic action object within a client creates all necessary data storage and provides visibility to all atomic action methods. An initialization method will read a configuration file and initialize the state table and other local data after construction of the atomic action object. A second configuration file will be used as stimulus to drive the operational scenarios.

A key aspect to this modeling approach is its extensibility. Additional methods can be added to the atomic action class definition as needed and the configuration file can reflect these additions to provide access to new atomic actions. Unfortunately this approach requires the user to develop C++ code and integrate this code into the simulator. This is addressed by the development of a user Application Programming Interface (API) for extending the functionality of the atomic actions class.

Appendix A: Modeling and Simulation of The Joint Battlespace Infosphere Scalability

Robert G. Hillman
James P. Hanna
Martin J. Walter

Air Force Research Laboratory
Information Directorate
Advanced Computer Architectures Branch
26 Electronics Parkway
Rome, NY 13441-4514
(315) 330-4029

Abstract

The Air Force is developing a concept known as the Joint Battlespace Infosphere (JBI) in order to achieve information superiority. The JBI builds on the Global Information Grid and will move the state-of-the-art in information management into information-centric warfare. JBI will address more than five million information objects, thousands of users and provide scalability challenges for currently available information management hardware and software systems and drive design requirements to create systems that are more capable.

JBI concepts will initially be created working with prototypes and will eventually culminate in fielded implementations. Important issues, such as bandwidth, connectivity, computational requirements and storage, information protection and assurance, must be addressed. Significant efforts will be required to implement JBI resources, such as network technologies and topologies, required to achieve JBI's stated objective to achieve warfare information superiority.

Key JBI resources are being modeled and simulated to identify, quantify, and resolve technology and topology issues influencing the prototyping, development, and deployment of an operational JBI. This simulation research will allow JBI developers to identify and mitigate programmatic risk early enough within the JBI seven-year development window to allow successful development and deployment of JBI. This paper will discuss the proof of concept assessment performed and the resulting development.

Introduction

Future DoD campaign strategies are being developed based on deployable distributed information management systems that are integral parts of the overall deployed footprint [1]. Combined, this integrated information system will provide assets and individual users with time-critical information required for their function during the crisis or conflict. This new deployable information enterprise will allow DoD forces to maintain information superiority over adversaries and react accordingly. The DoD will invest substantial resources over the next decade on the development of the distributed information enterprise infrastructure. To this end, the Air Force is developing a concept known as the Joint Battlespace Infosphere (JBI) [2] to achieve information superiority. The other Services are developing similar concepts, such as the Army's "Digitized Battlefield" [3] and the Navy's "Network-Centric Warfare" [4]. These service system-of-systems concepts expand to include Joint and Coalition Task Force deployment systems. These systems will move the state-of-the-art information management from the current network-centric warfare designs into information-centric warfare, thus providing raw and fused information to end users at presently unachievable quality and rates.

No single information architecture will be established to meet the operational requirements. Numerous architectures will be developed and established; the overall requirement that all implementations interact seamlessly.

An efficient commercial quality simulation environment is required to support the development of complex deployable information systems. This simulation research is establishing the necessary capability allowing developers to identify and mitigate programmatic risk early enough within the development to allow successful development and deployment of the associated systems. The simulation framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance. This development leverages previous DoD investments in event level simulation by building a generalized information architect simulator on top of SPEEDES [5,6].

The modeling framework will simulate the enterprise performance at the information level to identify, quantify, and resolve protocols, processes, and core functions influencing the prototyping, development, scalability and deployment of an operational enterprise. Simulations require "Challenge Problem" class resources to address more than five million information objects and hundreds of thousands of clients comprising a future information based force structure.

The design of a generalized simulation engine to support the modeling of a diverse set of distributed information systems posed considerable technical challenges. The conceptual framework, as seen in Figure 1, provides the user with the ability to separate the specification of nodal functionality from physical topology and communication protocols.

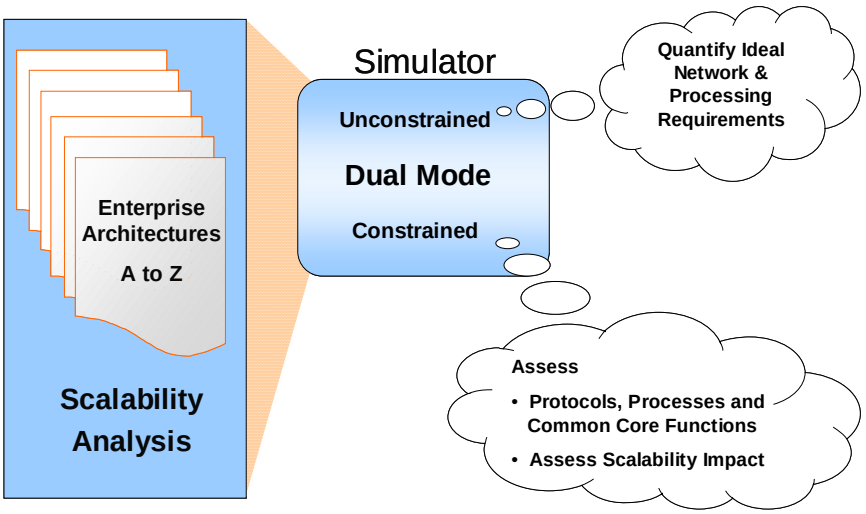

Figure 1. Conceptual Framework

Further, the framework provides the flexibility necessary to specify a broad set of nodal functionalities and physical topologies. The user can also specify operational scenarios necessary to drive the simulation and validate the model. The user is free from the necessity of writing code by defining model definitions for the information system enterprise in a declarative manner using configuration files.

Technical Approach

The goal of our framework design is to provide an underlying infrastructure and simulation engine that will allow users to model any information system enterprise in a declarative manner.

Through configuration files, the user specifies the system functionality, topology, and operational scenarios. The simulator reads these configuration files, builds the appropriate information system enterprise, and exercises the operational scenarios against the model. This technical approach is illustrated in Figure 2. The user is freed to focus on the design aspects rather than writing code to simulate and analyze the envisioned enterprise.

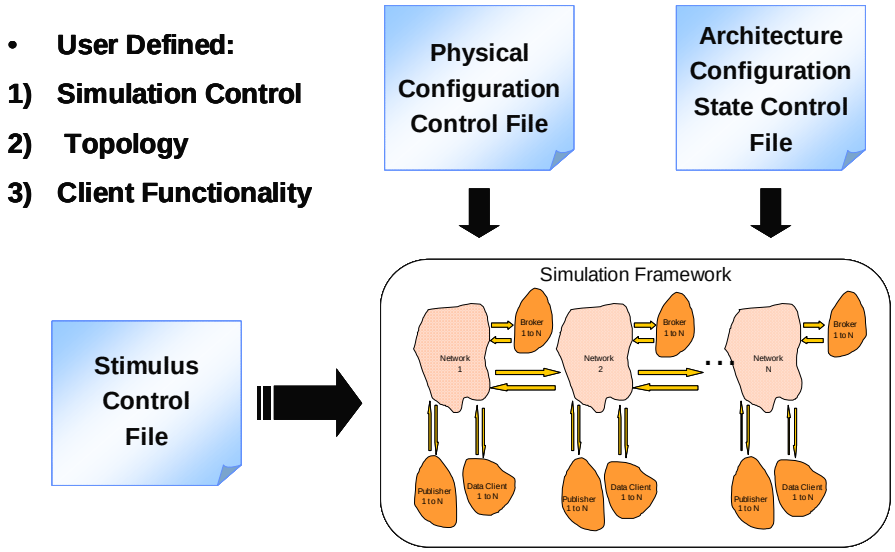

Figure 2. Technical Approach

The development was based on a nine-month study where prototypes of key aspects of the simulation framework were designed, coded, and evaluated. This evaluation phase was used to define the approach for the simulator design based on the lessons learned. The proof of concept coding was performed in the multiple environments; VHDL, JAVA and SPEEDES. This proof of concept phase reduces the project risk by verifying approaches and identifying the critical performance issues needing alternate solutions. This prototyping resulted in approximately 1200 lines of SPEEDES code written in C++.

A preliminary enterprise model exhibiting minimal behavior was used to validate the design approach for constructing an enterprise hierarchy. This model established a reference hierarchy between clients, providing some product (or functionality) to the enterprise and those that subscribe to (or consume) the information. The event driven communication protocol (how messages will be transmitted and received) between enterprise clients was used to evaluate a structured network model and communications protocol. A design review was performed to evaluate both the development concept for the simulation engine as well as simulation performance. A comparative analysis was made to assess the simulation and identify the strengths, weaknesses, and applicability of different approaches for the architecture analysis.

The SPEEDES-based parallel processing framework was selected as our target environment since it provides an ideal foundation for the development of a generalized modeling tool to support simulation of distributed information systems on a multiprocessor platform. The SPEEDES simulation framework provides processing power not available from a single processor. Applications that make use of SPEEDES

are typically time-constrained; too many events to process in a limited amount of time. SPEEDES allows the simulation builder to perform optimistic parallel processing on high performance computers, networks of workstations, or combinations of networked computers and HPC platforms.

Simulations were performed utilizing the preliminary architecture model. It was interesting to observe that with just over 500 clients, one million objects were processed and at 1000 clients 3 million objects were processed during initialization. The performance analysis, as seen in Figure 3, identified a major performance bottleneck in the network model used to connect the functional subsystems.

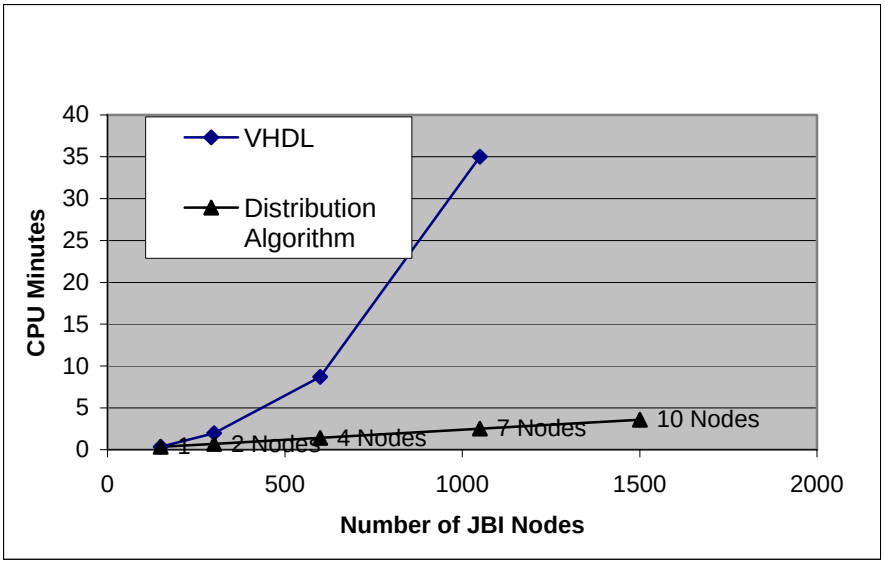

Figure 3. Performance Analysis

One thousand clients were simulated on a single CPU requiring 35 minutes to execute. A distributed simulation was performed on two nodes executing in 400 CPU minutes. As a result, a new prototype network model was coded utilizing a distributed approach resulting in orders of magnitude improvement for the distributed simulations. The resulting simulation employing up to 10 processor nodes to emulate 1500 clients executed in under 3 minutes. Simulations were expected to be highly event driven rather than CPU intense and the results confirmed that the event streams are definitely the overall performance-driving factor.

In order to further improve the HPC resource utilization, we evaluated several clustering algorithms to insure that clients with high event interactions would be constructed on common HPC nodes. As a result, the development optimization techniques have been evaluated for the simulation framework and the results are shown in figure 4. These optimization techniques were developed to balance the event distributions based on the simulation model.

The VHDL model employed an array of JBI nodes connected through a single network model. The nodes were of three types: PUBLISHERs, SUBSCRIBERs and BROKERs. All messages between the various nodes were transmitted through the network, which controlled the queuing of messages between the nodes, the timing, and the de-queuing of the messages. The number of publishers, subscribers and brokers were established via VHDL generics and were controllable at run-time.

The initial SPEEDES model mimicked the VHDL model, using a single network to connect all the nodes in the simulation. The SPEEDES simulations were performed on an SMP machine using one or two nodes. The network model was then modified so that it was composed of a network of sub-networks, as shown in Figure 2. This was then ported to a Beowulf Cluster for comparison. The number of nodes, their types, the number of subnets, and the distribution of nodes across the subnets were defined for the SPEEDES simulations at run-time using a topological configuration file.

A comparison of the performance of the various simulations is shown in Figure 3. The maximum number of nodes simulated was determined by the cpu-time used and by the capacity of the SMP with 1 node to complete the VHDL simulation. It can be noted that the VHDL simulation (which runs on a single node and has a single network) is substantially faster than the SPEEDES simulation even when running on the cluster and with a network described as a network of sub-networks.

It was noted that the single JBI network model was a bottleneck because all of the messages were passed through the network. Since the JBI network was placed on a single CPU, the JBI nodes not on that CPU communicated across the Beowulf Cluster's network to the JBI network model. This is evidenced by the relative performance of the SPEEDES simulations on the cluster with one node and with two nodes in Figure 3.

The network of sub-networks model was implemented using the SCATTER algorithm from SPEEDES for distributing the JBI nodes among the Cluster processors. There was a performance gain from this, but because of the preliminary nature of the enterprise model that was being simulated, where the inter-node communication was well behaved and well known, a more robust method for distributing the JBI nodes among the Cluster processors was needed. A distribution algorithm that grouped the nodes and their sub-network as described in a configuration file among the Cluster processors was developed. The results of that simulation, as compared to the VHDL simulation from Figure 3 is shown in Figure 4. The Figure shows that the distribution algorithm coupled with a network of sub-networks provided substantial performance improvements, even as compared to the fastest simulations from Figure 3, the VHDL simulation.

Additionally, for the simple test case, this algorithm provides nearly linear scaling with the number of JBI nodes, while the other simulations show a quadratic scaling relationship. For a more general application, where the node to node communication is less well known, this algorithm will provide performance enhancements, but the number of messages transmitted grows.

To further improve performance, a variation of the network modeling was examined. This variation incorporated the majority of the network model within each JBI node, and performing a direct node-to-node transmission. The network model was converted into a data collection and analysis model for the network traffic. A couple of techniques for updating the traffic information were examined. These included sending a message to the network for each message transmitted between the nodes. This doubles the amount of message traffic in the simulation. Another technique that was examined was for each node to compute its' contribution to the network during a time interval, and sending that information to the network at the end of each interval. The time interval is controllable through an external configuration file. This enables the user to set the granularity of the time-step so as to match the simulations needs while minimizing the overhead message traffic and its' detrimental effects on performance.

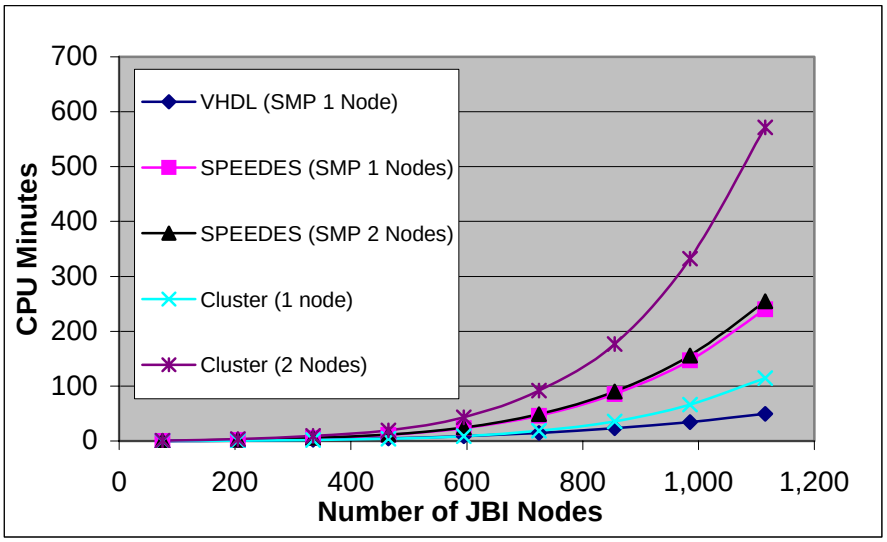

Figure 4. Optimized Distribution

Requirements Collection and Definition

Service implementations for the enterprise architectures will initially be created working with prototypes and eventually culminate in simulation of fielded implementations. The information enterprises will provide scalability challenges for currently available information management hardware and software systems and design requirements to create more capable systems. Each prototype and implementation will require numerous resources, including physical network implementations, as a basis to support design, development, and operational efforts. Important issues such as computational

requirements, storage, information protection and assurance, bandwidth, and connectivity must be addressed. Significant effort will be required to assess and evaluate these potential resource requirements. The distributed information enterprise must be simulated based on the operational behaviors of the members in conjunction with the physical constraints of the resources.

We are collecting and identifying all simulation framework requirements and feeding these requirements into the development of the core simulator.

1. Functional behavioral requirements will be extracted from prototype enterprise implementations that are being developed.
2. State machine representations of the behavioral functions that are to be imbedded in the information clients will be developed.
3. Physical resource characteristics will be identified for the underlying computer and network topologies from which model instantiations can be developed.

Client Functional Capabilities

Distributed information systems consist of a collection of autonomous clients interconnected via a network. Each client provides some product (or functionality) to the enterprise, and perhaps subscribes to (or consumes) one or more enterprise products. The specification of system topology is rather straightforward. The specification of individual client functionality, however, is not so simple. A given enterprise model may consist of thousands of clients distributed across tens or even hundreds of network domains. Client functionality cannot be achieved through static parameter instantiation; it requires a more sophisticated mechanism.

Our approach is to provide the mechanism to specify the behavior of each client as a state table. This state table will be a local client data structure built during the initialization phase of the simulation and will be based on an XML configuration file. The states enumerated in the state table will be callable methods that define atomic actions. Each state in the table may also consist of pre- and post- conditionals, as well as parameters necessary to fully define the atomic action desired. The order and aggregation of a given set of atomic actions will define the functionality of the client. Each client will iterate through its state table when it receives an activation event. The state pre-conditional may be used to determine whether a given state is active during the given cycle. The state post-conditional may be used to trigger events or set pre-conditionals of subsequent states. This approach will allow a broad set of functionality to be specified for clients.

All atomic actions are defined as SPEEDES public C++ class methods. Data types and objects, including conditionals, are defined using C++ container classes that are inherited by the atomic actions class. There is a one-to-one mapping between the elements specified in the XML configuration file and the methods defined in the atomic actions class. Instantiation of an atomic action object within a client creates all necessary data storage and provides visibility to all atomic action methods. An initialization method will read an XML configuration file and initialize the state table and other local data after

construction of the atomic action object. A second XLM configuration file will be used as a stimulus to drive the operational scenarios.

One technical challenge to this approach is the formation of a complete enough set of atomic actions that would allow for the specification of a diverse set of functionality. This goal may not be completely achievable. A library of actions will be developed based on a selected set of core capabilities that must be supported by the Air Force JBI initiative and selected enterprise applications. Several implementations of the JBI publish-subscribe scenario currently exist and will provide a good starting point for atomic action specification in the initial year.

A key aspect to this modeling approach is its extensibility. Additional methods can be added to the atomic action class definition as needed and the XML configuration file can reflect these additions to provide access to new atomic actions. Unfortunately this approach requires the user to develop C++ code and integrate this code into the simulator. A fallback approach would be to develop a user Application Programming Interface (API) for extending the functionality of the atomic actions class. Both approaches will be analyzed to determine the optimal solution.

Object Level Network Model

The interconnection network defines the bandwidth (or capacity) and latencies (or delay) of the connections between clients. Creating collections of clients with specific processing capabilities and networking characteristics can be achieved through instantiation with static parameters. Multiple network domains are modeled by associating sets of node clients with a given network domain and defining bandwidth and latency as a function of domain locality. In this approach, aggregating multiple network domains is achieved simply by partitioning the enterprise by domain association. Again, this can be achieved through instantiation with static parameters using an XML configuration file.

Modeling the information flow, among the clients in an enterprise containing many thousands of processing members in order to analyze associated enterprise scaling issues, must be performed utilizing model abstraction. The abstract models of the network fabric and the data packages will be constructed. This model abstraction, which will be part of the simulator framework, will represent the network described by high-level abstract characteristics, such as the latency and bandwidth of an equivalent link between any two nodes on the network. The functionality of the network and its time varying usage will be integrated with the simulation so that the effects of load, capacity, and interconnection can be adequately analyzed.

The simulated behavior of these abstract models must relate to the real world implementations. Therefore, these models must be calibrated and verified against a known reference, OPNET [7]. OPNET is the industry de-facto standard for emulating

detailed network behavior. The detail this network simulator provides can validate the scaling issues related to our abstract network model.

To calibrate and verify the abstracted network models developed for the JBISIM framework, network combinations with client clusters will be modeled with both OPNET and JBISIM. These simulations will establish the reference latency and bandwidth between processing clients in the network for use in the JBISIM Simulator. The abstract JBISIM models will be refined by back annotation using the OPNET simulations for identical networks with dozens of processing clients and a few subnets. Having developed and calibrated an abstract model for node-to-node latency and bandwidth on small and medium sized network segments, scaled simulations will be performed with thousands of nodes to validate the JBISIM network model. OPNET simulations of the same network configuration and data flow will be made. This iterative analysis process will calibrate the JBISIM network modeling, and verify its applicability to scaling issues, identifying limitations regarding precision and accuracy of the simulation. Discrepancies between the simulation results will be analyzed and used to adjust the precision and scalability issues of the simulator. Some variations are expected to remain but the goal is to develop the mapping algorithms so that a consistency within 10% is maintained.

Reference Architecture

All testing and evaluations of the simulation framework during the development process will initially be targeted at a reference enterprise architecture. This reference enterprise architecture will be developed based on system requirements collected. The intention is to develop a reference enterprise that will aggregate the behavioral requirements into a single test reference. This reference enterprise will be used to evaluate scalability and performance during the development process.

The reference enterprise will also be used in the demonstration phase. As the common model, it will be utilized for contrasting the scalability and performance results of the four target HPC systems. The reference architecture will be simulated on each system and statistics related to scalability and performance documented. Performance improvements addressing interconnect latency related to the pre- and post- optimization for SPEEDES will also be documented.

Conclusion

The proposed simulation capability will give the Air Force JBI team the ability to model key JBI related resources, identify, quantify, and resolve technology and topology issues influencing the prototyping, development, and deployment of the operational JBI enterprise. This simulation research will allow JBI developers to identify and mitigate programmatic risk within the JBI seven-year development schedule and help in the successful development and deployment of JBI enterprise. It is anticipated the Air Force Research Laboratory, Information Directorate, will support this simulation framework throughout the development years and beyond. During this period the simulator will be provided to any US Government agency requesting the framework to support their mission requirements.

References:

[1] United States Air Force Scientific Advisory Board, "Air Force Command & Control - The Path Ahead", Summary Report (Volume 1), SAB-TR-00-01, December 2000

[2] United States Air Force Scientific Advisory Board, "Report on Building the Joint Battlespace Infosphere, Volumes 1 and 2," SAB-TR-99-02, December 1999

[3] Frankel, Michael S., "The 1994 Army Science Board Recommended Technical Architecture for the Digital Battlefield", Army RD&A Bulletin, December 1994

[4] Alberts, David S., John J. Garstka, and Frederick P. Stein, "Network Centric Warfare: The Face of Battle in the 21st Century", Washington, DC: National Defense University Press, 1999.

[5] J. Steinman, "Scalable Parallel and Distributed Military Simulations Using the SPEEDES Framework," Elecsim 95, 1995.

[6] Gary Blank, Jeff Steinman, , "Design and Implementation of the High Performance Computing RTI for the High-Level Architecture", Proceedings Simulation Interoperability Workshop, Spring 2000

[7] Irene Katzela, "Modeling and Simulating Communication Networks - A Hands-on Approach Using OPNET", Prentice Hall, ISBN 0-13-915737-9, 1998

Appendix B: SPEEDES for Distributed Information Enterprise Modeling

James P. Hanna, Robert G. Hillman
hannaj@rl.af.mil, hillmanr@rl.af.mil

Air Force Research Laboratory, Information Directorate, Advanced Computer Architectures Branch, 26 Electronics Parkway, Rome, NY 13441-4514

ABSTRACT

The Air Force is developing a Distributed Information Enterprise Modeling and Simulation (DIEMS) framework under sponsorship of the High Performance Computer Modernization Office Common High Performance Computing Software Support Initiative (HPCMO/CHSSI). The DIEMS framework provides a design analysis environment for deployable distributed information management systems. DIEMS establishes the necessary analysis capability allowing developers to identify and mitigate programmatic risk early within the development cycle to allow successful deployment of the associated systems. The enterprise-modeling framework builds upon the Synchronous Parallel Environment for Emulation and Discrete-Event Simulation (SPEEDES) foundation. This simulation framework will utilize "Challenge Problem" class resources to address more than five million information objects and hundreds of thousands of clients comprising the future information based force structure. The simulation framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance. SPEEDES provides an ideal foundation to support simulation of distributed information systems on a multiprocessor platform. SPEEDES allows the simulation builder to perform optimistic parallel processing on high performance computers, networks of workstations, or combinations of networked computers and HPC platforms.

Keywords: Information enterprise, enterprise modeling, distributed simulation, high performance computing.

1. INTRODUCTION

Future DoD campaign strategies are being developed based on the concept of deployable distributed information management systems that are integral parts of the overall deployment footprint [1,2]. This integrated information system will provide assets and individual users with time-critical information required for their function during the crisis or conflict. This new deployable information enterprise will allow DoD forces to maintain information superiority over adversaries and react accordingly. The DoD will invest substantial resources over the next decade on the development of the distributed information enterprise infrastructure.

The Air Force is developing a concept known as the Joint Battlespace Infosphere (JBI) [3] to achieve information superiority. The other Military Services are developing similar concepts, such as the Army's "Digitized Battlefield" [4] and the Navy's "Network-Centric Warfare" [5]. These systems will move the state-of-the-art in information management from the current network-centric warfare designs into information-centric warfare, thus providing raw and fused information to end users at presently unachievable quality and rates. Further, individual Military Services integrated information enterprise architectures must employ system-of-systems concepts to expand their application to include Joint and Coalition Task Force deployment systems as well.

No single information architecture can or will be established to meet these operational requirements. Numerous architectures will be developed and implemented. These architectures are expected to continually morph to reflect the most current technology trends through their lifetime. The overall requirement is that all implementations interact seamlessly. This proposed three-year effort leverages previous DoD investments in event level simulation by building a generalized information architect simulator on top of SPEEDES [6,7].

Military Service implementations of the enterprise architectures will initially be created working with prototypes and eventually culminate in simulation of fielded implementations. The information enterprises will provide scalability challenges for currently available information management hardware and software systems and design requirements to create more capable systems. Each prototype and implementation will require numerous resources, including physical network implementations, as a basis to support design, development, and operational efforts. Important issues such as computational requirements, storage, information protection and assurance, bandwidth and connectivity will be addressed. Significant effort will be required to assess and evaluate these potential resource requirements. The distributed information enterprise must be simulated based on the operational behaviors of the members in conjunction with the physical constraints of the resources.

This project is part of the HPCMO/CHSSI System-of-Systems portfolio developing scalable software that emphasizes the integration of autonomously operating weapon systems into a dynamically controlled information network or System-of-Systems (SOS). In recent years, the Department of Defense (DoD) has recognized that weapon systems operating autonomously provide a less than optimal solution to our national security problems. Information processing nodes in the network will fuse information from other nodes to provide a relevant battlespace view to friendly participants. Due to the complexity of these systems, development and implementation of this capability will rival any other difficult development performed by DoD.

Testing of SOS will require simulations more complex than any developed to date. The tracking of interaction between hundreds of thousands of players, complex weapons systems, and environmental models while merging physics with information theory can only be achieved through the aggressive application of high performance computing and networking technologies. The complimentary applications in this portfolio address virtual system-of-systems characteristics ranging across the multiple domains. These efforts include: (1) Prototyping and Scalability Assessment of Distributed Information Enterprise, (2) Virtual Communication Links, and (3) Virtual Prototyping and Accelerated Testing of Systems. The portfolio efforts will all utilize SPEEDES as the underlying simulation framework and will be interoperable.

An efficient parallel, portable, flexible, scalable, and commercial quality simulation environment is required for the development of complex deployable information systems. The research reported here will develop and implement this capability. The DIEMS simulation framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance on the scale necessary to meet the aggressive requirements of DoD information enterprises.

1. ARCHITECTURE

The DIEMS architecture is intended to provide the framework, mechanism, and semantics to support the modeling and simulation of a wide variety of information enterprise functionality. The design of a generalized simulation engine to support the modeling of a diverse set of distributed information systems poses considerable technical challenges. The modeling framework provides the user with the ability to separate the specification of nodal functionality from physical topology and communication protocols. Further, the framework provides the flexibility necessary to specify a broad set of nodal functionalities and physical topologies. The user may specify the system communication and operational scenarios necessary to drive the simulation and validate the model. Most users will be free from the necessity of writing code by defining model definitions for the information system enterprise in a declarative manner using

configuration files. The modeling framework will simulate the specified information enterprise performance at the information level to identify, quantify, and resolve protocols, processes, and core functions as part of the prototyping, development, and scalability assessment of an information enterprise (see Figure 1).

The DIEMS interconnection network defines the bandwidth (or capacity) and latencies (or delay) of the connections between clients. Creating collections of clients with specific processing capabilities and networking characteristics can be achieved through instantiation with static parameters. Multiple network domains are modeled by associating sets of clients with a given network domain and defining bandwidth and latency as a function of domain locality. In this approach, aggregating multiple network domains is achieved simply by partitioning the enterprise by domain association. Again, this can be achieved through instantiation with static parameters using a configuration file.

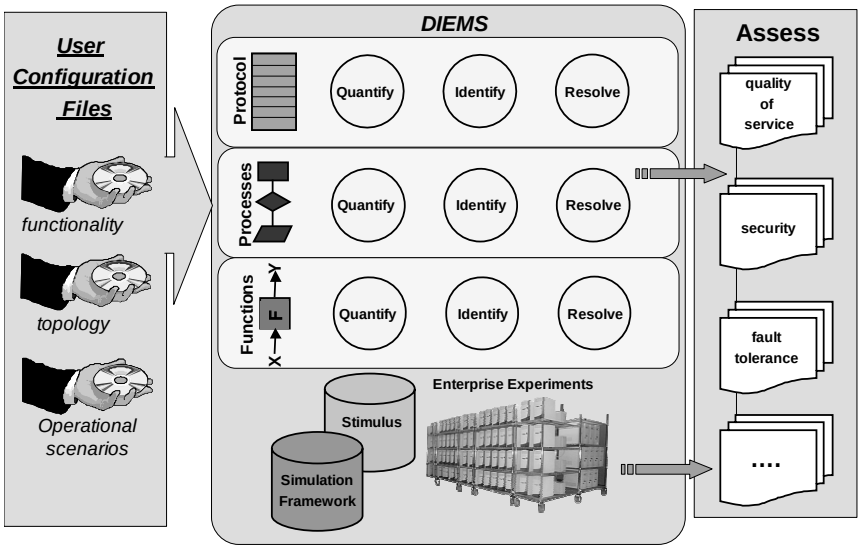

Figure 1. DIEMS Modeling Framework.

Distributed information systems consist of a collection of autonomous clients interconnected via a network. Each client provides some product (or functionality) to the enterprise, and perhaps subscribes to (or consumes) one or more enterprise products. The specification of system topology is rather straightforward. The specification of individual client functionality, however, is not so simple. A given enterprise model may consist of thousands of clients distributed across tens or even hundreds of network domains. Client functionality cannot be achieved through static parameter instantiation; it requires a more sophisticated mechanism.

Our architectural approach is to provide the underlying mechanisms to allow users to specify the behavior of each client as a process table. This process table will be a local client data structure built during the initialization phase of the simulation and will be based on a configuration file. The processes enumerated in the table are callable methods that define atomic actions. Each process in the table may also consist of pre- and post- conditionals, as well as parameters necessary to fully define the atomic action desired. The order and aggregation of a given set of these processes will define the functionality of the client. Each client will iterate through its process table when it receives an activation event. The process pre-conditional may be used to determine whether a given process is active during the given cycle. The process post-conditional may be used to trigger events or set pre-conditionals of subsequent processes. This approach will allow a broad set of functionality to be specified for clients.

All atomic actions are defined as public C++ class methods. Data types and objects, including conditionals, are defined using C++ container classes that are inherited by the atomic actions class. There is a one-to-one mapping between the elements specified in the configuration file and the methods defined in the atomic actions class. Instantiation of an atomic action object within a client creates all necessary data storage and provides visibility to all atomic action methods. An initialization method will read a configuration file and initialize the state table and other local data after construction of the atomic action object. A second configuration file will be used as stimulus to drive the operational scenarios.

A key aspect to this modeling approach is its extensibility. Additional methods can be added to the atomic action class definition as needed and the configuration file can reflect these additions to provide access to new atomic actions. Unfortunately this approach requires the user to develop C++ code and integrate this code into the simulator. This is addressed by the development of a user Application Programming Interface (API) for extending the functionality of the atomic actions class.

2. SPEEDES

The SPEEDES-based parallel processing framework provides an ideal foundation for a generalized modeling tool to support simulation of distributed informative systems. SPEEDES innate support for parallel processing across a wide variety of platforms, including SMP and distributed memory machines, establishes the means by which the DIEMS framework can make use of HPC to address large complex distributed information enterprise models. The native message passing and event infrastructure capabilities in the SPEEDES framework provide a robust communications architecture that directly supports information enterprise modeling. The SPEEDES general-purpose PAR file interface, with associated parser and set methods, provides a flexible and capable interface for specification of simulation control, instrumentation, configuration, and scenario files. The SPEEDES framework also provides an extensive user-configurable proxy service that allows direct run-time observation and manipulation of internal simulation objects. In addition, SPEEDES provides numerous classes that support the storage and manipulation of data necessary to address complex distributed simulation requirements. Finally, SPEEDES provides a general purpose C++ API that enables the integration and interaction with any C or C++ code base. This capability enables DIEMS to integrate easily with external tools such as the APACHE Xerces XML Parser and the PostgreSQL object oriented database suite.

DIEMS is currently using the SPEEDES Shared-Memory/TCP communications implementation to support parallel processing on the AFRL HADES cluster. This interface provides the underlying inter-node communication mechanisms over TCP that enables SPEEDES to support shared memory multiprocessors such as Beowulf cluster supercomputers. The DIEMS framework will make use of the SPEEDES MPI communications library to support a wider variety of HPCs after the initial development cycle is complete.

The SPEEDES messaging capability is an ideal fit for modeling information enterprises. The binary tree class provided by SPEEDES is used to maintain ordered message queues at each DIEMS client and network node in the simulation. Asynchronous processes in each DIEMS client and network instance facilitate this communication through the SPEEDES messaging API. Message statistics and network bandwidth metrics are gathered by associating figures of merit functionality with the SPEEDES messaging and event methods.

The general purpose SPEEDES parameter (PAR) file parsing and processing classes provide a simple yet powerful mechanism for configuring the DIEMS simulator. There are three approaches that can be used to setup a given simulation run. The DIEMS simulation uses several configuration files to incrementally specify how the simulation is built. These files define simulation control characteristics, enterprise topology and processor characteristics, client functionality and behavior, client user application scenarios, and simulation instrumentation observation and control characteristics. In the simplest case DIEMS used PAR files to specify this “build” information. The files are parsed by SPEEDES and the appropriate simulation is built. The user can also choose to specify a set of XML files to be used to configure the simulation instead of using the PAR file interface. XML provides a universal interface based on the network-centric standards in use today. The XML interface allows the DIEMS simulator to interface with a variety third

party tools. The user may also choose to specify a SQL database from which the simulation is to be configured. There are numerous benefits derived from the ability to interface with SQL database systems. Two such examples are configuration flexibility and extensibility, and data reduction and analysis. Figure 2 illustrates the three simulation configuration interface methods available to the DIEMS user.

The SPEEDES user-configurable proxy service permits the observation and control of internal simulation objects and data during simulation run time. This service allows all relevant DIEMS data to be visible through a SPEEDES "proxy server" while the simulation is running. Through this proxy server external programs can query the DIEMS simulation for data relevant to some set of observation criteria or to calculate some relevant system metric without incurring the processing or analysis overhead within the simulator. This capability enables the development of classes of analysis tools that serve to evaluate and quantify the enterprise performance and quality of service characteristics without interfering with the simulation. Since these analysis tools are external to the simulation the proxy interface can also be "turned off" without any consequence to the simulation.

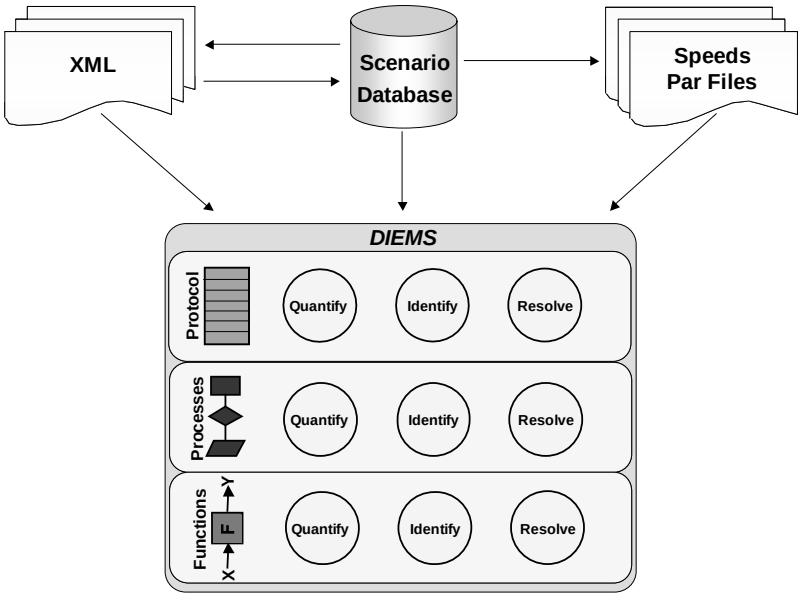

Figure 2. DIEMS Simulation Configuration Interfaces.

The C++ API provides the capability to integrate any existing C or C++ code base with the SPEEDES framework, and supports the object-oriented design and programming paradigm and supports code reuse. The standard C++ interface allows for portability and provides the user with the freedom to define and control the simulation and execution semantics for distributed information enterprise modeling and simulation.

3. CHALLENGES

Instrumentation and load balancers will be built into the SPEEDES framework to improve DIEMS simulation efficiencies. To do this, static and dynamic techniques will be identified, described, and evaluated. One or more of these techniques showing the most promise will be used to modify the SPEEDES simulation framework for the best performance for a specific enterprise architecture.

SPEEDES produces diagnostic files that support this development. For example, the Global Virtual Time (GVT) Statistics file records a number of metrics for each CPU, including CPU utilization, rollbacks, local virtual time (LVT), events and messages processed, and time spent in the various SPEEDES time management modes. Other diagnostic files include incremental event data CPU use, and a "flying trace," which records an event-by-event chronology. In a high event, low computation load environment, it will be important to also record which objects interact with each other most frequently so they can be nearest neighbors.

Currently, SPEEDES simulation objects can be distributed either singly, as a card deal, or in a block decomposition. Under Warsim 2000, the Army is funding a SPEEDES development effort for an optimization scheme that will place simulation objects on specified processors based on compute load. The DIEMS project will leverage that work by utilizing the same simulation object distribution scheme. Post processors will be developed to analyze the object passing messages and design an allocation scheme that will place DIEMS's event-intensive simulation objects on optimal processors.

The desired load-balancing capabilities will be develop in several steps. Insight into the load balancing needs can be gained by running variations of the DIEMS application on SPEEDES without load balancing, and collecting diagnostic output data on events, messages, and communication time intervals. This is an iterative process, where diagnostics are used to indicate one or more balancing techniques. One or more balancers (i.e. new logic for assigning events to CPUs) are designed and implemented and the simulations re-run to determine performance improvement.

This balancing step involves using the outputs of a simulation run to automatically select a load balance (if more than one is available) and enter all required parameters for subsequent runs. For example, if DIEMS is called on to perform a quick what-if-analysis, diagnostics from the first parametric run could be used to automatically implement balancing for the following run. Another application would be to use the diagnostics to rebalance a simulation when the simulation gets bogged down and then re-start the run with the new distribution.

Automatic load balancers can only go so far in determining optimal distribution of the simulation objects. In complex, extended scenarios, event profiles may change during a single run. A different approach is needed in cases where averaged balancing techniques do not support localized run efficiency requirements. For these cases, dynamic balancing techniques will be developed that optimize performance on a continuing basis. The dynamic techniques will impose restrictions on the design of simulation objects that will be automated for the placement of ghost behavior.

4. CONCLUSIONS & FUTURE WORK

The SPEEDES distributed simulation engine provides an excellent development platform for the DIEMS simulation framework. The capabilities and interfaces provided by SPEEDES enable the development of the DIEMS general-purpose enterprise modeling and simulation environment. The flexibility afforded by the standard C++ interface and built-in PAR file support, messaging system, and proxy interface, addresses all of our design and development requirements and allows us to satisfy all DIEMS program objectives.

One long-term objective is the integration of DIEMS into a complete end-to-end Global Information Enterprise simulation (GIESim) framework that focuses on communication technology as the integration medium for predictive warfare based information systems. Simulation objects need to be established for assessing the communication link, network management, and information flow. The GIE framework will use the High Level Architecture (HLA) publish and subscribe activities. It will also establish the capability of examining the problem from a top-down approach by interfacing simulation frameworks such as DIEMS with other M&S toolkits. This federated simulation framework will support GIE simulation across model abstraction layers. It will also support a bottom-up approach to segmenting the GIE and performing requirements analysis. These predictive simulation results will be fed back into the concept generation processes for future R&D refinement and evolution.

5. ACKNOWLEDGEMENTS

The authors would like to acknowledge the significant contributions to this project made by Mr. Martin J. Walter (Air Force Research Laboratory, Information Directorate, Advanced Computer Architectures Branch) and Mr. Robert L. Vienneau (ITT Industries, Advanced Engineering and Sciences Division). We would also like to acknowledge the contributions to this project made by Mr. Lou Concha of the Air Force Research Laboratory, Manufacturing Technology Directorate.

6. REFERENCES

[1] Owens, William A., "System-of-Systems: US' Emerging Dominant Battlefield Awareness Promises to Dissipate the 'Fog of War'", Armed Forces Journal International, January 1996

[2] Grimes, Vincent P., "Army Building Digital Foundation for Future: Electronically Connecting Combat Units Promises Dramatic Increase in Punch", National Defense 79, December1994

[3] United States Air Force Scientific Advisory Board, "Report on Building the Joint Battlespace Infosphere, Volumes 1 and 2," SAB-TR-99-02, December 1999

[4] Frankel, Michael S., "The 1994 Army Science Board Recommended Technical Architecture for the Digital Battlefield", Army RD&A Bulletin, December 1994

[5] Alberts, David S., John J. Garstka, and Frederick P. Stein, "Network Centric Warfare: The Face of Battle in the 21st Century", Washington, DC: National Defense University Press, 1999.

[6] J. Steinman, "Scalable Parallel and Distributed Military Simulations Using the SPEEDES Framework," Elecsim 95, 1995.

[7] Gary Blank, Jeff Steinman, , "Design and Implementation of the High Performance Computing RTI for the High-Level Architecture", Proceedings Simulation Interoperability Workshop, Spring 2000

[8] Bob DuCharme, "*XML: The Annotated Specification.* The Charles F. Goldfarb Series on Open Information Management" The Definitive XML Series from Charles F. Goldfarb. Upper Saddle River, NJ: Prentice Hall, ISBN: 0-13-082676-6, 1999.

[9] Irene Katzela, "Modeling and Simulating Communication Networks - A Hands-on Approach Using OPNET", Prentice Hall, ISBN 0-13-915737-9, 1998

[10] Hillman, Robert G., Hanna, James P., Walter, Martin J, "Modeling The Joint Battlespace Infosphere" Proceedings of the 6th International Command and Control Research and Technology Symposium, June 2001.

[11] Hanna, James P., Hillman, Robert G., Walter, Martin J, "Modeling and Simulation of The Joint Battlespace Infosphere Scalability" Proceedings of Summer Computer Simulation Conference, July 2001.

Appendix C: Analysis of a JBI Pub/Sub Architecture's Infrastructure Requirements

James P. Hanna, Martin J. Walter, Robert G. Hillman, Lois D. Walsh, PhD.
hannaj@rl.af.mil, walterm@rl.af.mil, hillmanr@rl.af.mil, walshl@rl.af.mil

Air Force Research Laboratory, Information Directorate
Advanced Computer Architectures Branch
26 Electronics Parkway, Rome, NY 13441-4514

ABSTRACT

A Distributed Information Enterprise Modeling and Simulation (DIEMS) framework, presently under development, is applied to the analysis of a Joint Battlespace Infosphere (JBI) Pub/Sub architecture's infrastructural requirements. This analysis is an example of one methodology that can be employed utilizing the DIEMS framework. This analysis capability permits the information systems engineer to ensure that the planned JBI architecture deployment will provide the required information exchange performance on the infrastructure provided. This paper describes the DIEMS framework including its application in constrained and unconstrained resource utilization modes. A JBI architecture is evaluated in the context of a representative operational scenario on one infrastructure. The simulator's unconstrained resource mode is employed to identify the architecture's ideal operational requirements and in turn identify potential resource limitations. The constrained simulation mode is employed to evaluate the potential choke points in relation to the architecture's performance. The results identify the infrastructure changes required so that the specific JBI architecture will achieve the required operational performance.

Keywords: Information enterprise, enterprise modeling, distributed simulation, high performance computing.

INTRODUCTION

The Joint Battlespace Infosphere (JBI) will provide the next generation information management infrastructure necessary to achieve and maintain information superiority over all adversaries in future DoD campaigns. The JBI concept was developed by the Air Force Science Advisory Board to address the broad range of information management challenges in joint military environments in a cohesive and extensible manner [15]. The JBI addresses all aspects of complex information management systems including information storage, retrieval, dissemination, control, reliability, and availability, with the goal of also addressing multi-level security. The units of information processing and transfer within the JBI are known as Information Objects (IOs). In the JBI, IOs from multiple and disparate sources are manipulated to create knowledge through the application of the core JBI concepts of Publish, Subscribe, Query, and Transform. This sophisticated integrated information management system will provide consistent, timely,

and appropriate situational awareness to all echelons from the war fighter in the field to the Joint Forces Commander.

The Air Force Research Laboratory, Information Directorate, has developed an initial JBI implementation that is based on the Jini service oriented architecture [3]. This JBI architecture is known as the Jini PUB/SUB implementation. Jini was designed to provide a distributed computing environment that enables networked systems to interact with each other at a network plug-and-play level. Networked systems may share services between each other by joining a Jini "community." Jini defines a set of protocols by which members of a given "community" provide, discover, and access these shared services. This implementation relies on the Java Remote Method Invocation (RMI) services of Jini to provide the infrastructure upon which the Jini JBI is built.

In the Jini PUB/SUB architecture, publication services are provided by a client known as the Publisher Adapter (PA). This client registers the intent to publish one or more IO with the Jini Lookup Service (LUS). Information describing the content and specific attributes of the IO is also provided at the time of registration. This information, known as metadata, is used by the LUS to uniquely identify the IO. A client known as a Subscriber Adapter (SA) follows a similar process to register the intent to subscribe to a particular type of IO with the Jini LUS. Here too metadata is provided during registration and the LUS preforms a match operation on the metadata to determine whether there exists PAs that satisfy the request of a SA. If the LUS determines that one or more PAs match the metadata registered by the SA, the LUS responds to the SA with the RMI proxy of all of the matching PAs. The SA may then invoke the RMI proxy(s) of the PA(s) to begin receiving subscriptions.

DIEMS

To assess detailed architectural trade-off and scalability factors relevant to the existing Jini PUB/SUB architecture and to support the rapid prototyping and evaluation of emerging and future JBI implementations AFRL/IFTC has developed a flexible and powerful enterprise information architecture simulation framework. This simulation framework makes use of High Performance Computing (HPC) technology to provide the efficient, parallel, portable, flexible, scalable, commercial quality simulation environment required for the development of complex deployable information systems. The Distributed Information Enterprise Modeling and Simulation (DIEMS) framework will be capable of assessing deployment aspects such as security, quality of service, and fault tolerance on the scale necessary to meet the aggressive requirements of DoD information enterprises [7][8][9].

The DIEMS simulator provides the framework, mechanism, and semantics to support the modeling and simulation of a wide variety of information enterprise architectures and implementations. The system defines topology, client functionality, and use-case scenarios that stimulate and drive the simulation separately through configuration files. One configuration file specifies the information enterprise topology and interconnectivity, as well as the compute resources and capacities of platforms that

comprise the hardware infrastructure. A second file specifies the placement of individual JBI clients (SAs, PAs, LUSs, etc.) onto the hardware resources that make up the hardware infrastructure. Of course, individual hardware platforms may host any number of JBI clients. A third configuration file associates the information objects, duration of publication, rates of publication, etc. with specific JBI clients specified in the previous file. The DIEMS system reads these files, elaborates the specified enterprise architecture, instantiates clients on platforms, allocates tasks (publications and subscriptions) to specific clients, and executes the enterprise simulation. Simulation traces are recorded along with a variety of user-specified information and statistics. A number of analysis and display tools have also been developed to aid in the visualization and analysis of the simulation results. These tools can be used to assess the simulation as it progresses or after the fact to post process the results.

DIEMS allows the user to simulate the information enterprise using either an unconstrained network model or a constrained network model. The unconstrained model provides an idealistic representation of network resources in the sense that communications between platforms and across domains are not in any way bandwidth limited. In the unconstrained mode the simulation allows the user to assess resources that a particular architecture will consume for a given JBI configuration and use-case scenario set without regard to physical limitations such as bandwidth, latency, CPU speed and processor memory. The constrained network model provides a more realistic representation of network resources by enforcing bandwidth limitations and calculating realistic message latencies. The constrained mode of operation allows the user to assess performance issues as they relate to a given enterprise architecture. Further, this assessment is made in the context of the processing and network traffic loads that are characteristic of the particular configuration and use-case scenarios of JBI applications running on the given infrastructure.

The network models underlying the constrained mode operation of DIEMS are an abstraction of the real-world network. DIEMS does not model networking at the packet-protocol level. Instead message latencies and bandwidth utilization are calculated based on message size and the current utilization and capacity of a given platform's network interface. These calculations are achieved by using transfer functions that characterize message latencies across the network. There are three cases that must be modeled to accurately address realistic network configurations. These characteristic transfer functions are being developed specifically for DIEMS by AFRL/IFS in Dayton OH. The first transfer function addresses small localized networks and is the result of detailed OPNET network simulations that analyzed various network configurations, topologies, routers, interfaces and protocols across the spectrum of network loading. This forms the basis for calculating latencies for messages that are transmitted within a geographically localized network. The second transfer function addresses medium-scale “campus” network configurations. This is the result of larger, more complex detailed OPNET simulations and is correlated with institutional or organizational network traffic loads that might be seen in corporate offices or universities. The third transfer function addresses messages that are routed through large global networks, such as the Internet. This is the result of research from the CAIDA research group who determined the approximate

number of hops between any two points on the commercial Internet [11]. These three characteristic transfer functions allow large heterogeneous, geographically dispersed information architectures to be modeled efficiently and at a reasonable level of fidelity to support physical infrastructures upon which a JBI could be expected to be deployed.

JBI NOTIONAL ARCHITECTURE

A conceptual scenario of an operational mission [13] was developed by the JBI team to evaluate the application of emerging JBI technologies and their potential impact on mission centric information management and exchange. This scenario does not reflect current operational reality but rather JBI enabled possibilities and is based on existing procedures and CONOPS. The operational scenario was developed based on a Time Critical Targeting (TCT) mission where timely and reliable delivery of the information objects is essential. A TCT is defined as a time sensitive target with an extremely limited window of vulnerability or opportunity where the prosecution of the target is critical to ensure successful execution of the Joint Force Commander's goals. Typical examples of TCTs include mobile theater ballistic missile (TBM) systems, mobile surface to air missiles (SAM), and tanks. Other, non-traditional TCTs also exist; an allied pilot downed in enemy territory is but one example. The key factors to the success of any TCT operation are the information acquisition and data exchange processes. For the purpose of this paper the specific scenario details have been abstracted to convey only the higher level application view.

The following assumptions were made in the development of the scenario:

- An integrated task force is available and responsible for the scenario execution.
- The assets that can be used to execute the scenario are all within the information enterprise.
- A centralized operations center is the lead facility which plans, tasks, and controls asset operations.
- A wide variety of ISR are available to provide intelligence on emerging time critical information.
- IP Communication protocols are being utilized over the varying communication systems to provide the information objects.

An Additional project goal was to assess, from a third party perspective the difficulty in understanding and developing a simulation based on the required XML configuration files established for the framework. Applying the case study we evaluated the user aspects of constructing the enterprise at the Alpha release of the DIEMS framework.

The JBI Enterprise Architecture for the case study was developed in three phases. The first phase developed the "Topology Configuration" which required mapping of the physical resources with the main emphasis on the number of computers and the network

interconnect required for each user or actor. Resources were specified and established based on the information objects which were required by the scenario based on publication, subscription, or query attributes. A computer platform was allocated for each information data object. For example, generation and publication of an Air Tasking Order would be assigned to one computer platform, and the generation and publication of Land Tracks another. Subscriptions and queries objects were assumed to be generated from the publications that were already established or from a common computer used for either subscriptions or queries. In total, the "Topology configuration" file defined the domains, the computer network interfaces and the number of computers required. Figure 1. illustrates a conceptual view of the network described by the "Topology Configuration" process.

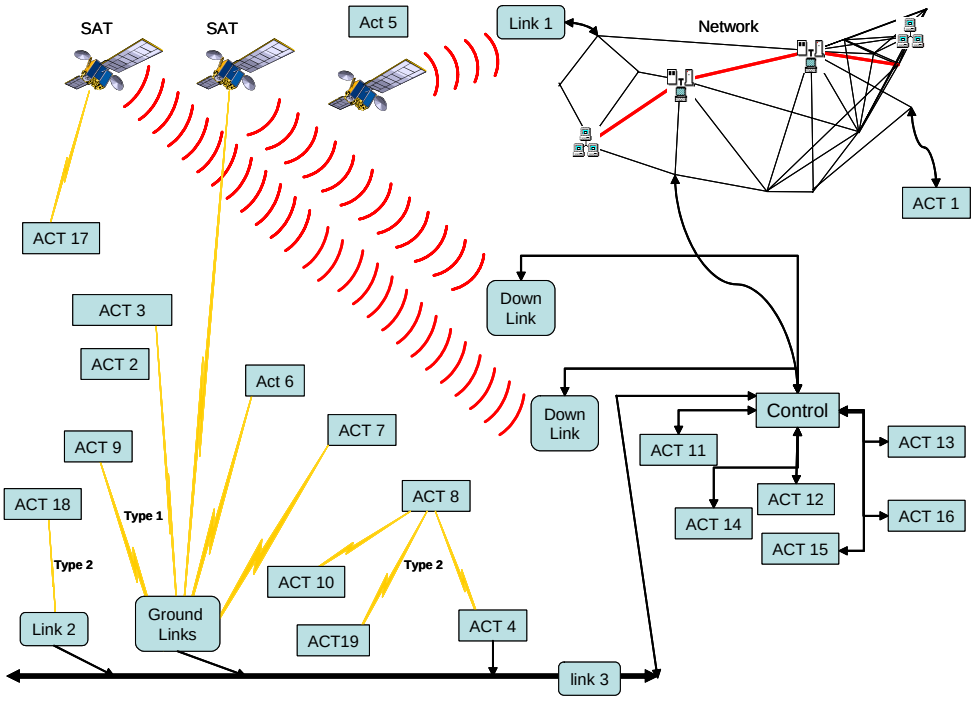

Figure 1. Network Topology

The "Behavior Configuration" defines the application or purpose (publish, subscribe, or query) of each computer on each domain as defined in the "Topology Configuration" file. Since we chose to allocate one computer for each major application function, this process was straight forward. An alternate approach would be to assign multiple application clients to common computer and process UIDs tracked when assigning (the functional operations) scenarios to the processes. The main task of this phase was to decide the type of behaviors that would be utilized to construct the enterprise. These behaviors are selected and constructed from a library of DIEMS user-selectable built-in functions. For this study, behaviors that emulate the Jini Pub/Sub implementation were employed.

The “Scenario Configuration” defines the data objects, payload size, and production rates for each scenario element. This file also defines the insertion times, start and end times of the scenario element to be used in creating the stream of operational data. Ideally the JBI TCT Case would follow the scenario setup and should flow easily from the previous files. To test this, a user that was not familiar with the underlying DIEMS infrastructure instantiated the configuration files. The user experienced a great deal of difficulty in developing these configuration files due to the lack of a glossary defining terms used in DIEMS. Many terms were vague and misleading as to their meaning, since the XML tag names were based on the developer’s perspective rather than that of the user. As a result, the XML name tags were changed to improve clarity.

Since the scenario didn’t address the JBI behaviors or the placement of those behaviors, the actual choice is left entirely up to the user establishing the enterprise. The placement of the publishers, subscribers, and query behaviors are defined by the information object definitions and therefore the placement is straight forward. There was one choice however related to the airborne sensors. These resources could be modeled from the downlink perspective where the sensor system and the ground system could be instantiated as a single object. An alterative approach, and the one chosen, was to instantiate objects and JBI behaviors for each sensor resource. This choice resulted in a more complex simulation but allowed for the inclusion of the actual RF communication links in-use today.

The first enterprise instantiated for the scenario also established a single object broker within the Air Operations Center. We realized that it was not sufficient to have one centralized broker for the entire scenario but we were interested in observing the operational impact. This choice resulted in a complete information overload within the broker domain and completely consumed all available resources. A second enterprise was established using multiple brokers. This approach had a significant impact on the overall performance of the enterprise.

The information enterprise simulated in this study included 66 network domains, 84 platforms, 7 different communication formats, and 5 different behaviors to support the Jini Pub/Sub architecture.

SIMULATION

The DIEMS simulator was applied to the Notional JBI architecture in the unconstrained mode of operation. This mode does not constrain the transmission rate of data between platforms, but allows the user to determine the communication resources that would ideally be required by the architecture.

Figure 2 displays a sampling of the communications traffic on the Notional Architecture. Traffic from the link in Figure 1 identified as Type 1 is displayed. In the Unconstrained simulation, shown in the topmost curve, the horizontal line with solid boxes is the

capacity of the as-designed JBI Architecture on the enterprise topology. This scenario uses as much as 10 times the maximum capacity of the Type 1 connection.

The simulation was executed a second time with the simulator in the constrained mode. The simulation results are displayed in the lower curve in Figure 2, where the utilization is clearly capped and the duration of the messages is extended significantly.

The total number of messages transmitted during the simulation are summarized by JBI Message Type in Figure 3. The majority of the messages are reply subscription, which is the fulfillment of a subscription by a publisher to a subscriber. Also a substantial numbers of register updates are tallied. These messages are part of the Jini process used to keep the enterprise current with regard to the status of the JBI clients. Less than 2% of all messages are of this type.

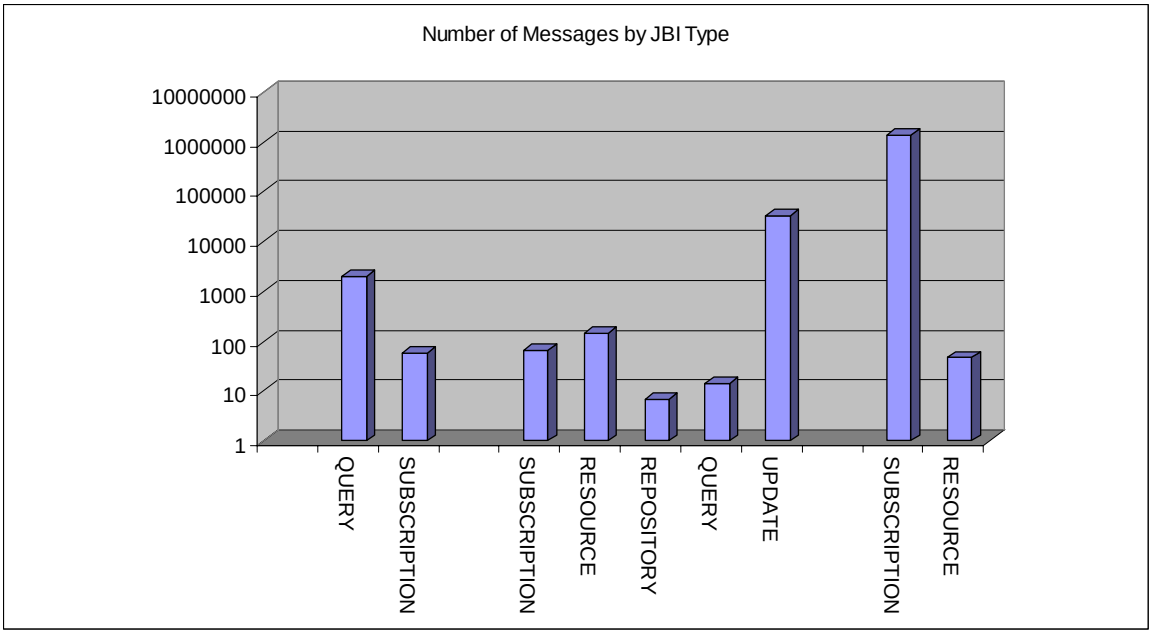

Figure 2. Network Traffic, Unconstrained vs. Constrained.

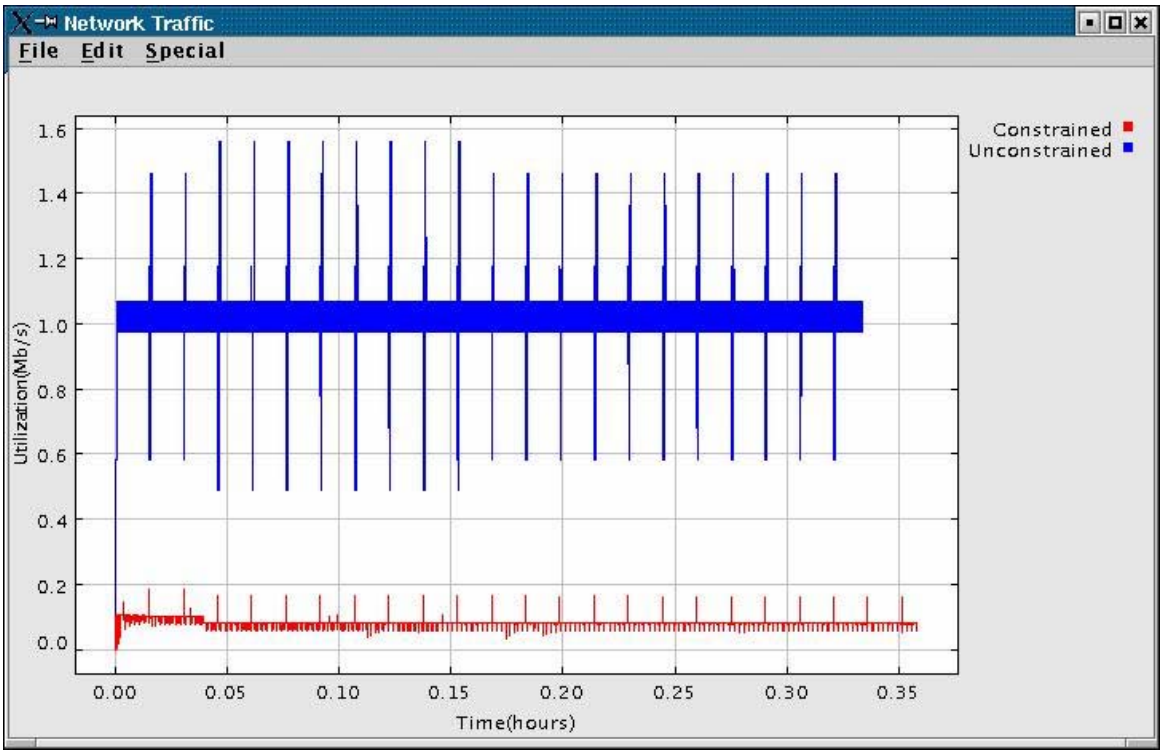

Figure 3. Number of Message by Type (left request; center register; right reply).

CONCLUSIONS

Parallel Discrete Event Simulation has been applied to the modeling and simulation of a complex information enterprise, hosting a Time Critical Targeting case study. The DIEMS simulation framework was described as well as how it was applied to the case study in its unconstrained and constrained modes of operation. The case study was of a modest size consisting of a few dozen network domains that were exchanging more than one million messages.

Analysis of the utilization of the network domains and message types revealed that some domains needed several times the network capacity for the expected traffic. In addition, the maintenance messages for Jini constituted a few percent of the total number of messages in the scenario.

The scenario that this case study is based on was developed from an operational view where specific simulation requirements were not addressed. In developing the simulation, it was quickly realized that the scenario textual description had several discrepancies and in some cases lacked the information necessary to perform the simulation. An example of this is that the existence and/or placement of JBI information

object brokers was not addressed at all in the document. This placed the burden of allocating these key resources on the configuration file developer. A better approach would be to perform the simulations as part of the scenario development as a means to validate the completeness of the operational perspective.

Implementation of the test set by the novice user ended up being more difficult than anticipated even given interactive sessions with the developers. Before another attempt is made, the underlying requirements need to be stabilized and communicated to the user. The process has clearly outlined the documentation requirement for a user manual. A moderate test case needs to be developed as part of the user guide and a step by step process outlined for generating the enterprise and scenario data. An API is also needed to document the system and establish the initial understanding of the overall framework to the user. The conclusion is that at this point only one of the developers familiar with the simulation framework and the JBI test cases has a chance of obtaining a useful result.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the significant contributions to this project made by Lt. Mike Dooley and his team at the Air Force Research Laboratory Information Directorate, Collaborative Simulation Technology and Applications Branch at Wright Patterson AFB. Team members include: Mr. Joel Dallaire and Mr. Jerome Reaper of Science Applications International Corporation (SAIC) Dayton Office.

REFERENCES

[1] Alberts, David S., John J. Garstka, and Frederick P. Stein, “Network Centric Warfare: The Face of Battle in the 21st Century”, Washington, DC: National Defense University Press, 1999.

[2] Gary Blank, Jeff Steinman, , "Design and Implementation of the High Performance Computing RTI for the High-Level Architecture", Proceedings Simulation Interoperability Workshop, Spring 2000.

[3] Combs, Vaughn T., Linderman, Mark, “A JINI-Based Publish and Subscribe Capability”, Proceedings of ITComm 2002.

[4] Bob DuCharme, “*XML: The Annotated Specification.* The Charles F. Goldfarb Series on Open Information Management” The Definitive XML Series from Charles F. Goldfarb. Upper Saddle River, NJ: Prentice Hall, ISBN: 0-13-082676-6, 1999.

[5] Frankel, Michael S., "The 1994 Army Science Board Recommended Technical Architecture for the Digital Battlefield", Army RD&A Bulletin, December 1994.

[6] Grimes, Vincent P., “Army Building Digital Foundation for Future: Electronically Connecting Combat Units Promises Dramatic Increase in Punch", National Defense 79, December 1994.

[7] Hanna, James P., Hillman, Robert G., Walter, Martin J, “Modeling and Simulation of The Joint Battlespace Infosphere Scalability”, Proceedings of Summer Computer Simulation Conference, July 2001.

[8] Hanna, James P., Hillman, Robert G., “SPEEDES for Distributed Information Enterprise Modeling”, Proceedings of the International Society for Optical Engineering Enabling Technologies for Simulation Science, April 2002.

[9] Hillman, Robert G., Hanna, James P., Walter, Martin J, “Modeling The Joint Battlespace Infosphere” Proceedings of the 6th International Command and Control Research and Technology Symposium, June 2001.

[10] Bradley Huffaker, Marina Fomenkov, Daniel J. Plummer, David Moore and K Klaffy, “Distance Metrics in the Internet”, CAIDA Research Group http://www.caida.org

[11] Irene Katzela, “Modeling and Simulating Communication Networks - A Hands-on Approach Using OPNET”, Prentice Hall, ISBN 0-13-915737-9, 1998.

[12] Owens, William A., "System-of-Systems: US’ Emerging Dominant Battlefield Awareness Promises to Dissipate the 'Fog of War’", Armed Forces Journal International, January 1996.

[13] Lt. Col. Robert E. Marmelstein, “Joint Battle Space Infosphere (JBI) Time Critical Targeting Use Case Version 1.2”, Joint Battlespace Infosphere Program Office Internal document.

[14] J. Steinman, "Scalable Parallel and Distributed Military Simulations Using the SPEEDES Framework", Elecsim 95, 1995.

[15] United States Air Force Scientific Advisory Board, "Report on Building the Joint Battlespace Infosphere, Volumes 1 and 2", SAB-TR-99-02, December 1999.

Appendix D: HPC Performance Analysis of a Distributed Information Enterprise Simulation

James P. Hanna, Martin J. Walter and Robert G. Hillman
Advanced Computing Technology Branch
Air Force Research Laboratory

ABSTRACT

Simulations of distributed information enterprises using the DIEMS framework [1] were performed on HPC clusters and SMP machines. The simulation results were analyzed with respect to the computation time involved in each process, the number of times processes were executed, the number of simulation rollbacks invoked during each simulation, simulation overhead, and the total wall clock time used to perform the simulation. The analysis identified several performance limitations and bottlenecks. One critical limitation addressed and eliminated was simultaneously mixing a periodic process model with an event driven model causing rollbacks. The second major factor limiting performance on cluster based systems was the cross-node communication. An optimization technique that exploits the knowledge of the publication and subscription paradigm of the information architecture being simulated was developed. This paper describes the simulation analysis, the modifications to the simulation models, the development of an optimization technique, and the impact of the code improvements on simulation performance.

INTRODUCTION

The Air Force is completing the development of a Distributed Information Enterprise Modeling and Simulation (DIEMS) framework under sponsorship of the High Performance Computer Modernization Program Common High Performance Computing Software Support Initiative (HPCMP/CHSSI). The DIEMS framework is a simulation system to analyze DoD deployable distributed information management systems. DIEMS establishes the necessary analysis capability allowing developers to identify and mitigate programmatic risk early within the development cycle to allow successful deployment of the associated systems [2][3][4].

The DIEMS simulator provides the framework, mechanism, and semantics to support the modeling and simulation of a wide variety of information enterprise architectures and implementations. The system defines topology, client functionality, and use-case scenarios that stimulate and drive the simulation separately through configuration files. One configuration file specifies the information enterprise topology and interconnectivity, as well as the compute resources and capacities of platforms that comprise the hardware infrastructure. A second file specifies the placement of individual JBI clients (SAs, PAs, LUSs, etc.) onto the hardware resources that make up the hardware infrastructure. Of course, individual hardware platforms may host any number

of JBI clients. A third configuration file associates the information objects, duration of publication, rates of publication, etc. with specific JBI clients specified in the previous file. The DIEMS system reads these files, elaborates the specified enterprise architecture, instantiates clients on platforms, allocates tasks (publications and subscriptions) to specific clients, and executes the enterprise simulation. Simulation traces are recorded along with a variety of user-specified information and statistics. A number of analysis and display tools have also been developed to aid in the visualization and analysis of the simulation results. These tools can be used to assess the simulation as it progresses or after the fact to post process the results.

The development and simulation of a JBI notional architecture identifying infrastructural requirements was reported in [5]. This paper reports on the analysis of the performance impediments that were uncovered by the application of DIEMS to this atypical example of a distributed information enterprise. This atypical example was designed to stress the DIEMS framework and illuminate potential performance issues. We also discuss modifications made to the simulation framework to address these performance impediments and present results showing the performance improvements achieved.

PERFORMANCE ANALYSIS

Having successfully completed the functional and performance requirements for the DIEMS Alpha release, a concentrated performance optimization effort was executed. The optimization strategy was divided into three phases that were executed in sequential order, each leveraging the prior phase. The first phase was to profile the Alpha system and evaluate the individual functional blocks for performance. This phase included an investigation of the functional blocks that were consuming compute resources and an assessment of new algorithms or data structures to reduce the computational time. The second and third phases focused on reducing the rollbacks that were being generated within the SPEEDES parallel simulation framework. The second phase redesigned the platform processing algorithm to replace the inefficient time driven algorithm with a purely interrupt driven algorithm, reducing unnecessary processing and minimizing the effects of induced rollbacks. The third and final phase involved optimizing object placement on specific CPU nodes in order to reduce node to node interactions by clustering high traffic platforms on common CPU nodes.

PHASE I

SPEEDES provides a built-in ability to instrument the simulation and analyze process performance. The SPEEDES instrumentation was used to profile the alpha system by recording total events and CPU time utilized for each functional block. The atypical, worst case test scenario was also generated to drive the simulation during this phase of the analysis. A significant aspect of this simulation configuration was that a network topology and scenario were established whereby communications were streamed continually between opposite ends of the enterprise. This scenario was atypical of a truly

distributed enterprise but established an extremely difficult test case for the purpose of our analysis.

Figure 1 summarizes initial execution analysis that required over 20 days of CPU time with a wall clock of 5.3 days. Initial analysis indicated that over ¾ of the CPU time was consumed within the SPEEDES framework which caused an initial concern relative to the architectural design used for DIEMS. This concern was later determined to be unfounded. We expected that the rollbacks were a significant contributor to the framework overhead. Another interesting observation was that almost 1/3 of the CPU time was consumed by a single event interface that simulated the platform to platform network routing.

Upon inspecting and analyzing the process routing algorithm it was determined that considerable time was being spent in evaluating the routing data structure. The algorithm was modified to include a caching function to eliminate the duplicative analysis. The overall result was a reduction in wall clock from 5.25 days to 8 hours. During the 8 hour run, the DIEMS core consumes 5.3 CPU hours with the SPEEDES core consuming 203 CPU hours. The code was next analyzed with respect to evaluating the event interfaces relative to rollback and the associated processing time.

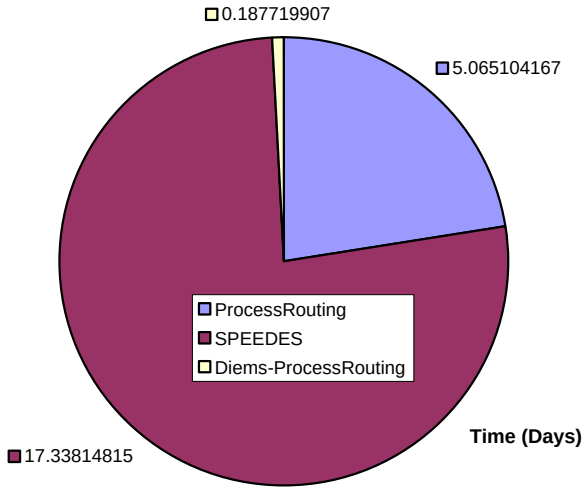

Figure 1. Total Processing Time

Figure 2 illustrates the rollback related process times with four interfaces being the major contributors; “NetRouterReciever”, “ExecuteRouting”, “ProcessRouting” and “Processing”. The first three functional interfaces establish the network interface of the platforms. The behavior emulates a switching router that facilitates a store and forward mechanism. The analysis identified that the store and forward mechanism was responsible for the rollbacks associated with these event interfaces. The network interface was modified combining the functions into a single interface that emulates a router performing a simple receive and forward function.

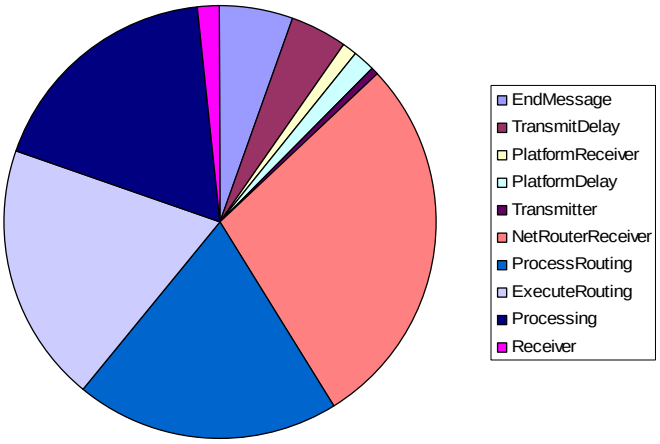

Figure 2. Rollback Relative Processing Times

Figure 3 illustrates the relative processing times of the top CPU intensive event interfaces at the completion of this first code optimization phase. The event execution times are now somewhat balanced with "NetRouterReciever" and "Processing" consuming 60% of the time, a reasonable state since they reflect the main functions, processing atomic behavior and routing the messages. At the completion of this phase the overall wall clock processing time was reduced from 5.25 days (126 hours) to 3.6 hours with rollbacks reduced from 91 million to 43 million.

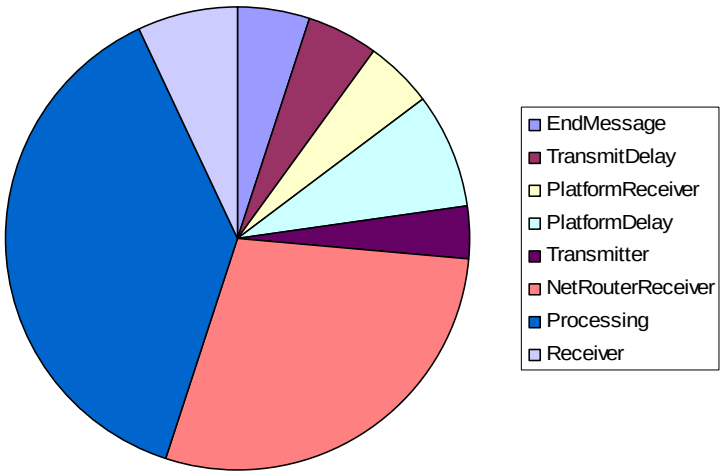

Figure 3. Final Relative Processing Times

PHASE II

The second optimization phase focused on redesigning the platform processing algorithm to reduce execution time and rollbacks. In this discussion the term platform is used to identify one of the virtual computer objects on the simulated network/enterprise. The entire simulation is comprised of interconnected platforms and network domains (or routers) given some specific enterprise topology.

The initial platform processing algorithm evaluated the task queue on each platform based on an interval timeout to schedule platform processing events. At each interval all events expected to occur before the next interval timeout were scheduled via the SPEEDES API call SCHEDULE_*SomeEvent*(). This API call enters the given event into the SPEEDES event queue for processing. On multiprocessor systems each node upon which the distributed simulation runs processes into the future independently of other processor nodes. When events that occur in the past are received by a processing node, the simulation must roll back. To achieve this, all events at the receiving node must be undone; the simulator on that node must reset time to accommodate the past event and then reprocess all of the events that were rolled back.

The judicial selection of an appropriate timeout interval is critical to minimizing the number of rollbacks the system must process. Rollbacks are very expensive computationally. Unfortunately, selecting the correct interval depends upon the granularity of the network traffic scenarios and the actual node placement of objects within the simulation. In the general case, no appropriate selection can be made.

To alleviate the problems associated with the interval timeout and to minimize simulator rollbacks an event driven approach to platform processing was devised. This new approach, albeit more complicated, has the advantage of assuring that at most a single event per platform will ever require reprocessing when a rollback occurs. The event-based algorithm evaluates the platform's task queue and schedules the current event for the given platform. Next it evaluates the task queue to determine when the next event occurs and schedules a wakeup event on itself. This process continues until there are no more events to schedule. This redesign of the platform processing algorithm reduced the number of rollbacks from 43 million to 22 million and decreased the wall clock processing time from 3.6 hours to 1.8 hours.

PHASE III

The final optimization phase focused on object placement and required the development of a more realistic operational scenario where information clusters existed between geographical local communities.

The DIEMS simulation objects, specified in the topology configuration file, are distributed across the HPC nodes using a decomposition algorithm. Platforms are distributed with their respective domains, and by default, domains are distributed in a

block method. The block method places the first ***n*** processes on the first HPC node, the next ***n*** processes on the next node, etc.

The message traffic between HPC nodes is strongly dependent upon which domains and platforms are on which HPC nodes and is also strongly dependent upon the details of the message traffic specified in the scenario configuration file. It is an oversimplification to assume that domains near each other in the topology numbering scheme will necessarily dominate the communications. In order to remove the uncontrolled character of the internode's communication, a scheme was developed to place domains that share the highest number of inter-domain messages on the same CPU. This controls and reduces the inter-CPU communication during the simulation and reduces the number of simulation rollbacks processed.

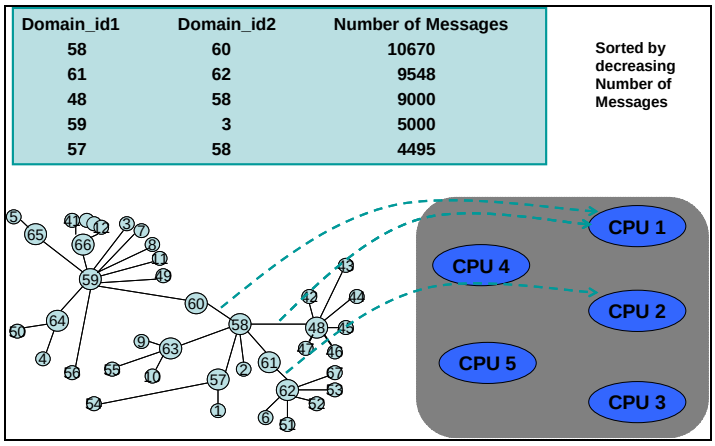

Figure 4. Simulation Object Placement Scheme

In order to place domains on the CPUs based on the message traffic, the number of messages between domains must be known. One method to obtain the inter-domain message traffic before executing the simulation, is to enumerate the messages. For deterministic protocols, such as Jini PUB/SUB, all messages can be known apriori, by examination of the configuration files.

Enumerating the messages has the additional advantage of enabling verification of the simulation, in that the simulation should produce the same messages, but distributed in time.

Enumerating the messages is based on knowledge of the publication and subscription paradigm being used. A message enumerator for the Jini PUB/SUB architecture was developed to provide a summary of the number of messages. The result is a list of the messages identified by the message Class, Detail and Data Type, received by each platform in the simulation. It also generates a summary list of the inter-domain message traffic, which can be represented by a set of nodes and edges, where the nodes are the domain IDs, and the "edges" are the inter-domain connections. The inter-domain message traffic list is ordered by the decreasing number of messages on each edge, as can be seen in Figure 4.

Placement of the domains and their associated platforms is performed by using the order that the domains appear in the list of "edges". Figure 4 shows an idealized multiprocessor with five CPUs, the topology configuration for the domains, and a sample "edge" file with domain IDs and the numbers of messages exchanged between the domains. The decomposion is accomplished by checking each domain on an "edge" for previous placement. If one domain on the "edge" has been placed, the other is placed on the same CPU. If neither has been placed, both domains are placed on the same CPU. After all of the domains have been placed on the CPUs an Object Placement file is written. DIEMS was modified to optionally read this file, and distribute the domains and their platforms accordingly during simulation initialization.

A variation on the Atypical example configuration used in [1] was employed to evaluate the simulation object placement with the DIEMS framework. This configuration consists of 67 domains and 152 platforms. The communication architecture employed is JiniPub/Sub. The domain connectivity is seen in Figure 4 and Figure 5, where the clusters are indicated by the ellipses, the domains are indicated by the numbered circles, and the inter-domain connectivity is indicated by the black lines. The circles not contained in any ellipse indicate a fifth cluster. Verification of the configuration and its' implementation in the DIEMS framework was accomplished by comparing the enumerated number of messages for each type, class and detail with the number obtained from the simulation output. More than 1.5 million messages were received during the simulation. This configuration has a strong grouping of the message traffic into five clusters, with about 3% of the communication between the clusters.

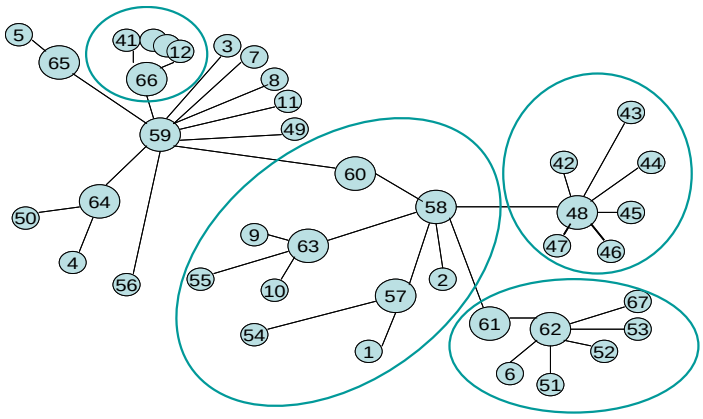

Figure 5. Communication Cluster

The configuration was simulated on an HPC cluster using differing numbers of CPUs. Figure 6 shows the time to completion for the simulations as the number of CPUs was varied. The figure shows two data sets; one for the default block mode object decomposition, and one for the message-based decomposition (BACH). As the number of CPUs is increased, the use of message-based decomposition results in improved time to completion as compared to block mode. Further, the use of block mode shows the usual upturn in completion time as the number of processors increases. This upturn is not apparent with the use of message-based decomposition.

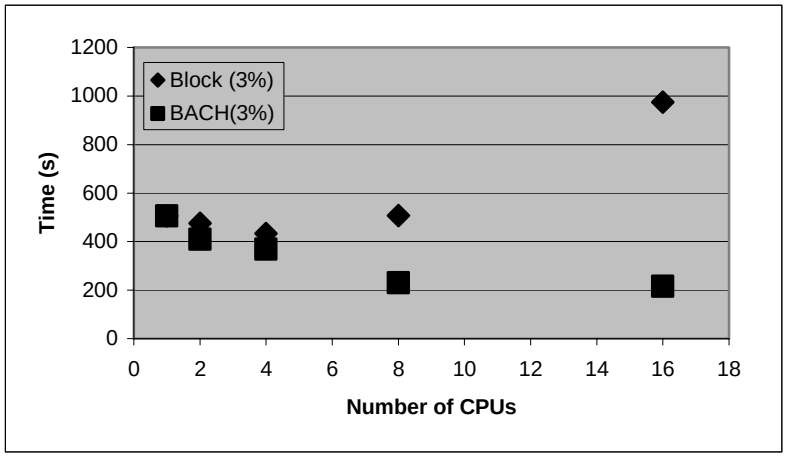

Figure 6. Simulation Time vs. object decomposition algorithms

The decomposition algorithm uses only the order and connections of the inter-domain message traffic. The degree of the clustering of the message traffic effects the placement, and the simulation performance. For cases where the inter-cluster message traffic is a larger percentage of the total traffic, the inter-cluster edges are placed earlier and the clusters may not be cleanly distributed among the CPUs.

Consider the case of a cluster of seven domains, domains 42 through 48 in Figure 5. For a simulation where the message traffic between domains 48 and 58 is greater than the traffic between 48 and any other domain in that cluster, domains 48 and 58 are placed on the same CPU, to reduce the number of cross CPU messages. The result is that those two clusters will reside on the same CPU. Even for modest percentages of inter-cluster message traffic, this can occur since the percentage is inversely proportional to the sum of the traffic among the cluster's domains, while the placement is based on a comparison of individual "edges."

SUMMARY

The Distributed Information Enterprise Modeling and Simulation (DIEMS) framework under sponsorship of the High Performance Computer Modernization Program Common High Performance Computing Software Support Initiative (HPCMP/CHSSI) and its performance enhancements have been described. This paper reported on the analysis of performance impediments that were uncovered in the application of DIEMS to an atypical worst case example of a distributed information enterprise. These limitations included excessive CPU time in ProcessRouting, extremely large numbers of simulation rollbacks, and uncontrolled inter-process communications, which aggravated the rollback problems.

These issues were addressed by implementing a caching algorithm for the ProcessRouting; developing an internal scheduler, which eliminates group rollbacks and

reduces the processing time on unnecessary updates; and by implementing an inter-process communication based simulation object placement algorithm.

These modifications reduced simulation run time for eight CPUs from more than five days to less than 25 minutes, reducing the simulation wall clock to 0.3% of its original value.

REFERENCES

[1] Hanna, James P., Hillman, Robert G., and Walter, Martin J. "Modeling and Simulation of the Joint Battlespace Infosphere Scalability" Proceedings of the Summer Computer Simulation Conference, July 2001.

[2] Hanna, James P., Hillman, Robert G., Walter, Martin J, "Modeling and Simulation of The Joint Battlespace Infosphere Scalability", Proceedings of Summer Computer Simulation Conference, July 2001.

[3] Hanna, James P., Hillman, Robert G., "SPEEDES for Distributed Information Enterprise Modeling", Proceedings of the International Society for Optical Engineering Enabling Technologies for Simulation Science, April 2002.

[4] Hillman, Robert G., Hanna, James P., Walter, Martin J, "Modeling The Joint Battlespace Infosphere" Proceedings of the 6th International Command and Control Research and Technology Symposium, June 2001.

[5] Hanna, James P., Walter, Martin J., Hillman, Robert G., Walsh, Lois D. PhD. "Analysis of a JBI Pub/Sub Architecture's Infrastructure Requirements.", Proceedings of the International Society for Optical Engineering Enabling Technologies for Simulation Science, April 2003.