

## AN ACCURATE, SCALABLE COMMUNICATION EFFECTS SERVER FOR THE FCS SYSTEM OF SYSTEMS SIMULATION ENVIRONMENT<sup>1</sup>

Rajive Bagrodia

Scalable Network Technologies

Steve Goldman

Integrated Defense Systems, Boeing Co.

Ken Tang

Scalable Network Technologies

Dilip Kumar

Integrated Defense Systems, Boeing Co.

### ABSTRACT

The Future Combat Systems (FCS) program is developing the FCS System of Systems Simulation Environment (FSE) to provide the “real world wraparound” to the FCS System of Systems Simulation Framework (S2F). A primary component of the FCS is the Communication Effects Server (CES) whose objective is to develop a flexible, scalable, and high-performance, packet-level, discrete-event simulator that will accurately portray the behavior of the FCS communications architecture to eventually support the live, constructive, and virtual simulations envisaged in the FSE. In particular, the CES is required to compute, in real-time, accurate *end-end latency* for every communication message sent over a wireless network in a FSE experiment.

This paper provides an overview of the CES that has been developed using the QualNet network simulator. It presents results on the performance of the CES for the simulation of large on-the-move communication networks in real-time.

### 1 INTRODUCTION

Network-centric military programs such as FCS rely on communications as a critical force multiplier. The tactical communication infrastructure will be provided via wireless, mobile, ad-hoc networks (commonly referred to as MANETs). The performance of such networks is substantially impacted by a number of factors including traffic load, mobility, terrain and environmental effects that might cause dramatic changes in link capacities, and hence on the end-to-end latencies and message completion rates.

The FCS System of Systems Simulation Environment (FSE) is being developed to simulate the operation of FCS systems. A key capability in the FSE is the communication effects server (CES) that can accurately and dynamically assess end-end performance of a traffic mix with Quality of Service (QoS) requirements. Legacy network simulations typically lack the mix of accuracy and speed to make such dynamic analyses possible in large scale wireless networks. Consequently, previous efforts have typically relied on a process that separates the analyses into distinct phases of connectivity analysis, network route computation, and throughput computations. However, the quality of a wireless link (which ultimately determines connectivity and end-end completion rates and latency) is time varying in nature due to the interplay between signal strength and interference and is affected by environmental factors as well as the actions of other communication devices in the vicinity. Thus, while separating connectivity analyses from link quality assessment and route selection algorithms, may reduce computational complexity, it will also introduce substantial inaccuracies in the computation of end-end performance which limits its utility. This is particularly true with the dynamic and adaptive nature of next generation wireless devices that incorporate protocols that exploit cross-layer interactions, software-defined radios, and smart antennas. The CES described in this paper provides a capability for simulating a communication architecture where connectivity, link quality, interference, routing, and Quality of Service effects are considered simultaneously in a state-based simulation model to provide accurate predictions of end-end performance metrics including delay, message completion rates, and packet loss rates.

Simulation and its use in predicting performance of computer and communication networks has a rich history. A number of network simulators have been developed, both as University research projects, and commercial prod-

---

<sup>1</sup> Approved for Public Release, Distribution Unlimited, TACOM 2 Aug 2006, case 06-173

ucts. Commonly used network simulators include GloMoSim[Glomosim], QualNet[QualNet 2003], NS-2[ns2], OMNET++[omnet] and OPNET[opnet]. As the complexity of the modeled system continues to increase, researchers have exploited parallel execution of the discrete-event simulation models to reduce its run-time. An active research community has developed around parallel discrete-event simulation (PDES)[Fujimoto 91], and particularly its application to network simulation [Bajaj 1999, Riley 2004, Nicol 2003]. A major focus of this effort has been to significantly increase the size of the simulated network by relying on the increased CPU and memory resources of the parallel architectures. The primary challenge for the CES was to leverage the progress made by the simulation and networking research community in simulation of large-scale wireless networks, while supporting its use in experiment contexts that required the simulation to run in hard real-time such that it could be interfaced with interactive live and virtual subsystems.

Rather than build the CES from scratch, an early architectural decision was to leverage the capabilities of the QualNet network simulator, developed by Scalable Network Technologies. QualNet is the commercial successor of GoMoSim[Xeng 1998], a simulation software originally developed using the PARSEC simulation language[Bagrodia 1998] at the UCLA Computer Science Department. Key design attributes of QualNet that influenced this choice include:

- Modular interfaces among protocol stack layers to support detailed, accurate models throughout the protocol stack, including the physical layer.
- Transparent parallel execution of network simulation models on diverse parallel architectures including shared-memory multiprocessors, distributed memory multi-computers, and cluster computers.
- Well-defined external interfaces to support interoperability (via DIS, HLA, or socket connections) with other software.

The need to support interactions with live and virtual articles imposed a hard real-time requirement on the CES. The ultimate objective of the FCS CES is to support hard real-time simulation of wireless networks with very large numbers of communicating devices at a sufficiently high level of accuracy that the simulation can interoperate with both physical test articles and human participants. Our results to date have shown that the QualNet-based CES can successfully simulate end-end communications generated in the operation of an on-the-move, wireless network with thousands of communicating elements, in transactional real-time. For transactional real-time simulation, each end-end message communication in the model must be simulated in less time than would be needed by the correspond-

ing physical network to deliver the corresponding message from its source to its destination. This paper focuses primarily on this transactional performance of the CES which allows us to evaluate how well the CES can serve as a communication modeling tool in real-time simulation systems in which physical hardware, software, or human-users are included in the experimental environment.

## 2 CES ARCHITECTURE

The objective of the CES is to compute *reachability* (i.e., does the message reach its intended destination) and *end-end latency* for every communication message that is sent between platforms in a FCS simulation experiment. Henceforth, we will use the term *communication* to mean a message that is exchanged between platforms. For every communication, the CES will take into account the impact of multiple factors on each communication to determine if and when the corresponding communication sent by the *source* platform is received by the *destination* platform(s). The CES simulates the transmission of not only the data traffic, but also explicitly incorporates the *control (or overhead) traffic* that is generated within the network to perform multiple activities including maintaining connectivity, calculating routes, electing access points, or gateways, etc. Various facets of the communication network that are simulated in the CES include:

- Packetization of the source message into packets(or fragments) based on specification of the maximum packet size specified for the transport protocol
- Transmission of the packets through every layer of the protocol stack from the source radio to the destination and including each intermediate device on a (possibly) multi-hop path from the source to the destination.
- The protocol level effects simulated by the CES include group formations, clusters or region formation; packet forwarding and dynamic route calculation; multicast group formation, acknowledgements etc.
- MAC layer protocol effects including contention, collision, time slot or channel allocation retransmissions, etc.
- Radio-level effects including frequency reuse, modulation, antenna properties, transmit power etc;
- Environmental-level effects including terrain, weather, foliage, jammers etc;
- Quality-of-service level effects including priority, scheduling and queueing disciplines, etc; and
- Simultaneous transmission of competing data and control traffic.

## 2.1 CES Operating Modes

The CES was designed to support simulation experiments in one of four different forms, respectively referred to as *live*, *virtual*, *constructive*, or *standalone* mode.

In the *standalone* mode, all inputs to the CES, including communications, are provided from pre-generated data stored in files. The CES is executed in an ‘as-fast-as-possible’ mode, and it may run considerably faster than real-time. This mode is also referred to as ‘logger file’ mode as any communications from an external simulator have been generated in an off-line mode and stored in a log file which is used to provide the message inputs simulated by the CES.

In the *constructive* mode, the CES is run together with a force-on-force simulator (e.g. OTB, OOS, Combat XXI), but the latter is run using a script with no human inputs. The force-on-force simulators typically use Computer Generated Forces (CGFs) to represent platforms on some battlefield. The communications simulated by the CES are generated primarily by the CGFs. The execution of the CES is constrained by the execution of the force-on-force or combat simulator. Although the execution of the CES may need to be synchronized with the execution of the combat simulator, there is typically no requirement for the CES to be synchronized with real-time.

In the *virtual* mode, the CES is executed with a combat simulator (e.g. OTB), such that there may be *dynamic* and *interactive* inputs sent by the combat simulator to the CES. The inputs include *communications* as well as updates that affect the position, damage state, and other attributes of a platform or a communication asset.

Lastly, in the *live* case, the components in the experiment include physical or hardware components that interact with the simulated components and the CES. As the CES receives inputs (including communications) from physical devices and might send outputs (including communications) to physical devices, it is necessary that the execution of the CES be synchronized with *real-time*.

The overall schematic of the CES is shown in Figure 1.

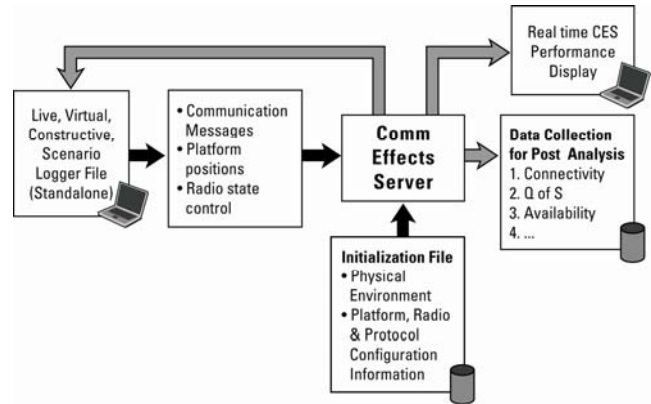


Figure 1: CES Architecture

As shown above, the CES takes two types of input information: initialization information which includes information about the physical environment including terrain; the initial position of each platform, and the number and configuration of *radios* on each platform. In addition to the initialization information, the CES dynamically receives new platform positions; communication messages which include source, destination, length, priority and related information; and updates to change the on/off state of a radio. Additional interfaces can be provided to dynamically send control information to alter the operating characteristics of a software-defined radio.

Two primary types of outputs are produced by the CES. First, for each communication, the CES computes the end-end completion and transmission latency. Second, over 150 different network and application level statistics are collected, which might either be displayed in real-time, or collected and stored for subsequent review and analysis. Additional metrics will be added to support emerging FCS Network Technical Performance Measures (TPMs) and Measures of Effectiveness (MOEs).

## 2.2 CES Modeling Framework

To meet the needs of the FSE, it was necessary that the CES architecture be able to model systems to a precise level of detail while delivering real-time performance. To maintain accuracy, the CES framework was developed using a layered approach similar to the TCP/IP stack architecture, as shown in the figure below. Detailed models of commonly used protocols have been developed, and the simple APIs defined between neighboring layers allow the rapid integration of new models. *The APIs are kept as close as possible to the operational network protocol stack, such that even operational code is easily integrated into QualNet with this layered design.* The operational code integration capability has been demonstrated at the transport layer by extracting the TCP model from the protocol code distributed with the FreeBSD operating system, and at the network layer by integrating the OLSR MANET

routing protocol code made available by INRIA. As the FCS operational code matures, it can also be directly integrated into the CES and effectively provide an emulation capability, for experimental instantiations where this level of fidelity is needed.

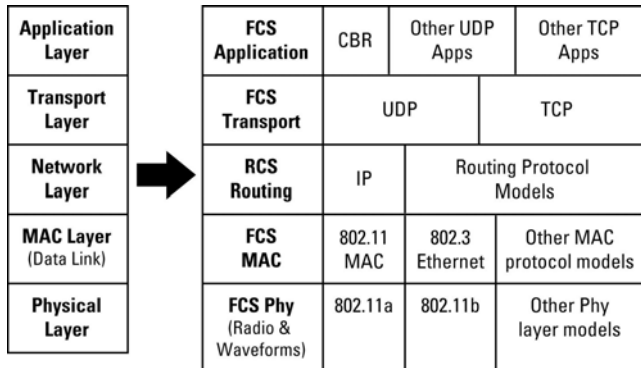


Figure 1 : FCS Communication Architecture Modeling Framework

The FCS communication architecture modeling framework and its mapping to the network protocol stack is shown in the above figure. In the framework, the FCS components are modularized into layers along the lines of the traditional ISO/OSI layering architecture. The FCS application layer is responsible for modeling traffic behavior that is characteristic of an FCS network. This includes accepting communications request from external networks. The FCS transport layer provides both reliable and unreliable services to voice, video, data and other types of applications. The hierarchical organization of the FCS architecture is handled by the FCS routing layer, which provides capabilities for *group formations*, *IP address change*, *dynamic group management*, and *routing*. Frequency, time and modulation selection, along with medium access, are the duty of the FCS MAC layer. The FCS physical layer models the physical characteristics of the corresponding radios and waveforms. Quality of Service mechanisms may be provided at each of the FCS layers. Mobility and node attrition are handled by separate modules that interface with the layers.

The CES supports a number of propagation models including terrain-integrated models like TIREM, and a number of path loss models that are critical to computing signal strength loss accurately, including the two-ray path loss model, together with other models that can add various environmental and communication effects including Doppler, multi-path fading, weather and other atmospheric effects. The CES models at the physical layer accurately account for signal strength attenuation due to path loss, interference, and accumulated noise from nearby radios.

### 3 REAL-TIME SIMULATION

As the complexity of the simulated system increases, the time taken to execute the simulation model can increase dramatically. For instance, other factors being equal, the time to simulate a wireless network can increase as a quadratic function of the number of nodes in the network. In constructive use contexts, although there is a general requirement for efficient execution of the simulation model, there is no hard requirement that the simulation complete within a pre-defined duration. However, in a live or virtual simulation, strict real-time requirements may be imposed on the execution time of the model to ensure its relevance for the experiment.

Henceforth, we use the term *physical* (or *real*) clock to refer to the clock used to measure physical time and the term *logical* (or *simulation*) clock to refer to the clock used in the simulation. In a discrete-event simulation, the logical clock is advanced in discrete intervals. We further use the term *simulation time* to refer to the amount of time that has elapsed in the simulation as measured by the simulation clock and *wall-clock time* to refer to the amount of physical time that has elapsed in executing the simulation. In general, the simulation time and the wall-clock time advance independently of each other; thus, it is entirely possible for a simulator to use 1s of wall-clock time to advance 10s in simulation time, or to consume 10s of wall-clock time to only advance 1s in simulation time.

Real-time performance has been defined in multiple ways; for instance see [Ziegler 2000]. A simple definition of real-time simulation is to require that the total *wall-clock* time needed to execute a model be no more than the total simulation time of the experiment. We refer to a simulator that meets this definition as a mean real-time (MRT) simulator. In a MRT simulator, on average, the events occurred as fast as or faster than real-time, although it is likely that some events took considerably longer than real-time. The second, more restrictive definition of real-time is a mean transactional real-time (MTRT) system. This definition of real-time indicates that the wall-clock time to execute an average transaction was less than its simulation time. That is, some transactions were processed faster than real time and some slower but on average all of the simulation transactions were real-time.

In interactive experiments, where physical components (and perhaps human participants) are interacting with simulated components, or in multi-paradigm environments [Zhou 2004], where subsystems are modeled using different modeling paradigms (analytical, emulation or even physical realization), it is necessary to impose a more strict requirement on the timeliness of the simulator. We use the term *transactional real-time* simulation to refer to a simulator that must satisfy the constraint that the wall-clock time required to simulate every *transaction*, say  $t$ , modeled by the simulator, be less than some *a priori* bound, say  $b_t$ .

In particular, a transaction may be defined as the end-end transmission of a message from a source vehicle in a platoon to a destination vehicle at the Tactical Operations Center. Similarly, the bound  $b_i$  may be specified to be the end-end transmission delay of the real world counterpart of the simulated message. This class of simulation is termed peak transactional real-time (PTRT). Clearly, MRT processes do not need to be MTRT since the MRT definition also includes any silence periods that do not contain active traffic (such as lightly loaded networks). PTRT is the most restrictive definition and is therefore the most difficult to achieve in simulation.

We assume that physical or virtual components in a FSE interface to the CES at the message level. Hence a transaction is the end-end delivery of a message; henceforth we use *transaction* and *message* synonymously. Two metrics are of primary interest: the latency of each transaction; and whether the transaction was completed (i.e., the message was delivered). If a transaction is not completed, it has infinite latency. All dropped messages are, by definition, PRT and the simulation only has to inform the interface when the packet has been dropped at some point.

Although it may be impossible to achieve PTRT for every message in a simulation (particularly as the network becomes large and activity more bursty), we define a metric to quantify the fraction of transactions that do meet the PTRT requirement:

$$R = \frac{\Delta t_{\text{wall-clock}}}{\Delta t_{\text{simulation}}}$$

For each completed transaction, the ratio  $R$ , referred to as the *Timeliness Ratio*, represents the fractional deviation of the wall-clock time from the simulated transmission delay for the corresponding message. If  $\Delta t_{\text{wall-clock}} \leq \Delta t_{\text{simulation}}$ , the ratio is equal to or less than unity and indicates real-time or supra-real-time performance for the corresponding transaction. If  $\Delta t_{\text{wall-clock}} > \Delta t_{\text{simulation}}$ , the ratio is greater than unity and indicates sub-real-time performance for the transaction (message). We use  $S$  to denote the Peak Transactional Real-Time Ratio for the experiment, which is the fraction of messages for which  $R$  was at most 1; the objective for a PTRT simulation is to have an  $S$ -value that is ideally 1.

## 4 PERFORMANCE RESULTS

This section presents a sampling of results to demonstrate the ability of the CES to provide transactional real-time performance in the simulation of wireless network with several thousand radios. As the focus of this paper is on the performance of the simulator, we ignore any issue dealing with the performance characteristics of the simulated network. For the experiments, each radio is modeled using a complete, high-fidelity protocol stack that includes mod-

els of the protocols specified for the radio at each of the transport, routing, MAC and PHY layers. The radios were distributed homogeneously in multiple regions, with each region covering a 7sq km area. The terrain description for a region is taken from DTED level 1 data of the corresponding area. The Irregular Terrain Model (ITM) is employed to calculate path loss based on the DTED terrain used in the simulation scenario.

The CES was executed with three different traffic profiles, and three mobility scenarios. For the traffic profile experiments, we consider traffic that is characterized as *low*, *medium* and *heavy* and includes a mix of UDP (unreliable transmissions) and TCP (reliable transmissions) sessions. The communication links are assumed to have a bandwidth of 2Mbps; the application traffic (or *offered load*) is varied such that the *low* load scenario constitutes 10% of the 2Mbps bandwidth per area, medium load represents 30% of the 2Mbps bandwidth (or 600 Kbps) per area, and high load denotes 60% of 2Mbps bandwidth (or 1.2Mbps) per area. Note that these percentages refer to *offered* load that excludes the control traffic generated by the network. It was observed that in the *heavy* traffic scenario described above, the resulting *carried load* in the network (which includes the data and control traffic) is sufficient to saturate the network.

Three distinct mobility experiments were executed, characterized by static (or no mobility), running speed (defined as 4mps or about 8 mph), and car speed (defined as 26.8mps or about 60 mph) mobility respectively. Note that all nodes are assumed to move at the average speed specified for the scenario and movements are constrained by the area of the sub-region within which a node was originally placed.

The hardware configuration that was used to run the simulation experiments is a 16 processor Blade Linux cluster. The Blade cluster has the following specifications:

- 8 nodes, each with a dual Opteron K8S Pro EATX motherboard
- 2 x AMD Opteron Model 246 (2.0GHz)
- 4GB PC3200 ECC Reg DDR (400MHz)
- Switch, SMC 8612T – 12 port 10/100/1000 managed switch
- SUSE Linux Professional 9.1 (64 bit) OS

In summary, for all simulations executed with different traffic loads and mobility patterns described above, the CES was able to achieve the hard real-time requirement PTRT simulation for all but a handful of the messages when using up to 16 processors of the blade architecture while simulating a wireless network with thousands of radios. We present a sampling of the results from our experiments.

Figure 2 depicts the runtime performance of the CES expressed in terms of the Mean Real Time metric dis-

cussed in the preceding section, for each of the three different traffic loads imposed on the network. Under all traffic profiles, we observe from Figure 2 that the percentage of real-time used is approximately 62%, indicating the ability of the CES to easily meet real-time constraints for wireless network simulations.

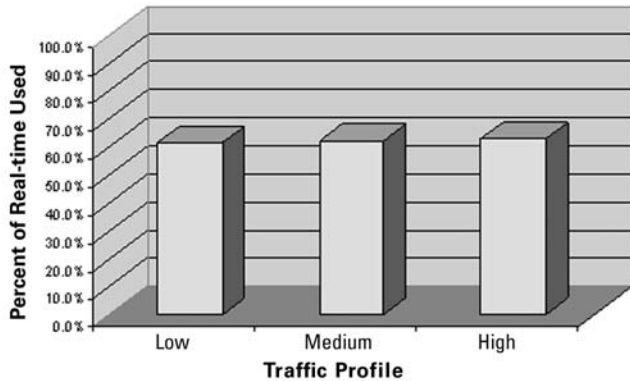


Figure 2: Percentage of real-time used for traffic profile experiments.

We now present the PTRT performance of the CES using a scatter graph. In Figure 3 and Figure 4, for the *low* and *high* traffic scenarios, the graphs plot the Timeliness Ratio (R), for each message whose successful delivery is simulated by the CES. It is clear that the PTRT ratio (S) for these experiments is very close to 1. The graphs also show that the timeliness ratio for each message is typically between 0.5 and 0.7 and even for the very few messages where the PTRT deadline is missed, the ratio is not significantly greater than 1.

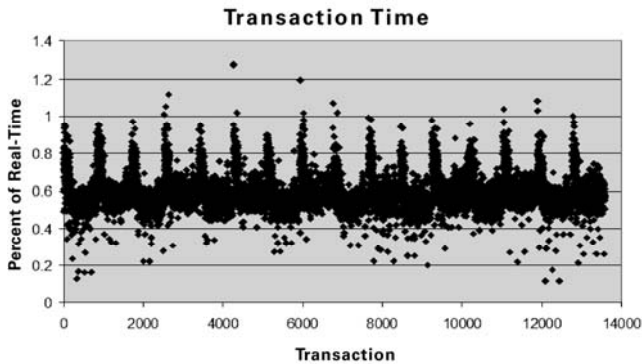


Figure 3: Per packet transactional real-time for low traffic profile experiments.

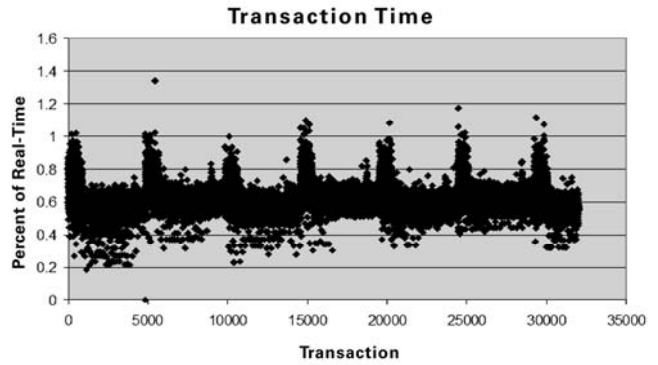


Figure 4: Per packet transactional real-time for high traffic profile experiments.

Figure 5 provides the mean transactional real-time ratio experienced by all the messages in the traffic profile experiments. The figure shows that under low, medium and high traffic profiles, the CES is able to obtain an average transactional real-time ratio of 0.61, 0.62 and 0.63 respectively; in other words the end-end simulation of an average transaction only requires 62% of the end-end transmission delay experienced by its physical counterpart.

Our last set of experiments present the impact of parallel architectures in improving the real-time performance of the CES. First Figure 6 plots the mean real-time for simulation of a large wireless network with low traffic and no mobility as a function of the number of CPUs used to execute the simulation; with 16 CPUs the CES is running 2x real-time.

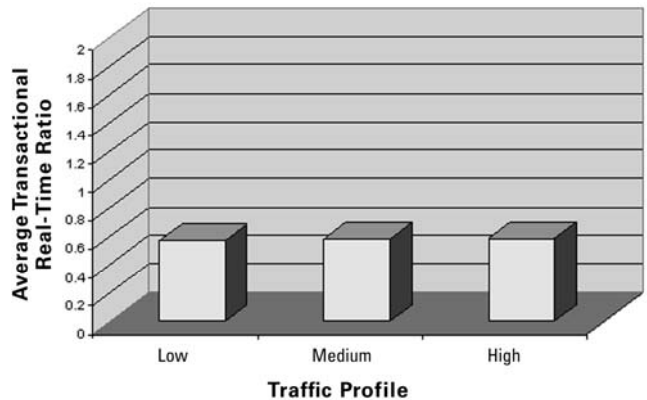


Figure 5: Average transactional real-time ratio for traffic profile experiments.

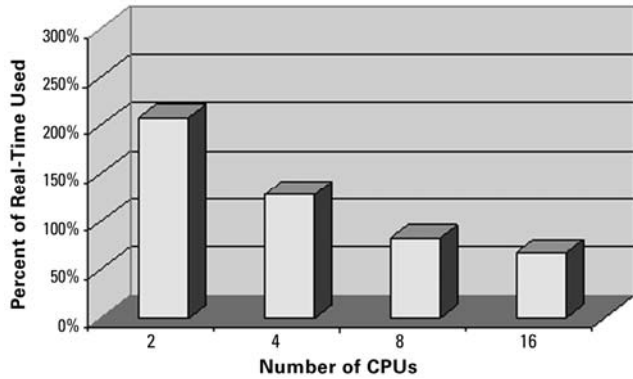


Figure 6: Real-time ratio for parallel experiments.

We demonstrate the impact of parallel simulation on improving the PTRT ratio for the experiments. From Figure 7, we see that out of over 32000 messages being processed, only a few of the messages were processed in transactional real-time using 2 CPUs. By utilizing 16 processors, the CES can meet the transactional real-time requirements for all messages within a minuscule fraction of a percent (Figure 8).

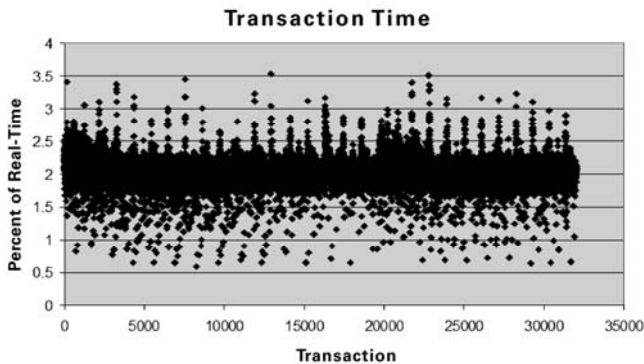


Figure 7: Per packet transactional real-time for 2 processor profile experiments.

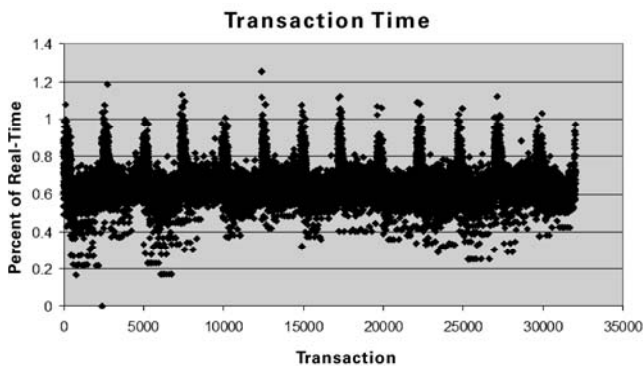


Figure 8: Per packet transactional real-time for 16 processor profile experiments.

## 5 CONCLUSION

A key requirement of the FCS Simulation Environment is the ability to accurately and dynamically simulate the impact of communication effects in the end-end performance of a FCS System of Systems. This paper presented an overview of the Communication Effects Server that is being developed for this purpose. A set of results were also presented that show the ability of the CES to achieve transactional real-time for nearly all messages in a wireless network with thousands of nodes under different traffic and mobility scenarios, using no more than 16 processors of a standard cluster computer.

## 6 ACKNOWLEDGEMENTS

We are grateful to Jeff Weaver of SNT for formalizing the notion of transaction real-time simulation, to Rich Meyer and Mineo Takai of SNT for help with multiple aspects of the parallel CES implementation, to Doug McKeon from CERDEC, Ft Monmouth for his participation in the CES prototype, and to Paul Watson and Kent Pickett of the US Army MSMO for their strong and continuous support of the CES project. Thanks to members of the CES project team at SNT and the LSI for contributions to the development of the models and systems reported in this paper.

## 7 BIBLIOGRAPHY

- R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, B. Park, and H. Song. Parsec: A parallel simulation environment for complex systems. *Computer*, 31(10):77–85, October 1998.
- Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Rajive Bagrodia Simulation of Large-Scale Heterogeneous Communication Systems," *Proceedings of MILCOM'99*, November 1999.
- R.M. Fujimoto, "Parallel Discrete Event Simulation", *Communications of ACM*,33(10), pp.30-53, 1990.
- David M. Nicol, Jason Liu, Michael Liljenstam, Guanhua Yan: Simulation of large-scale networks using SSF. *Winter Simulation Conference 2003*: 650-657.
- Ns-2. <http://www.isi.edu/nsnam/ns/>.
- QualNet User Manual v3.8. Scalable Network Technologies, Culver City, CA. <http://www.qualnet.com>.
- George F. Riley, Mostafa H. Ammar, Richard M. Fujimoto, Alfred Park, Kalyan S. Perumalla, Donghua Xu: A federated approach to distributed network simulation. *ACM Trans. Model. Comput. Simul.* 14(2): 116-148 (2004).
- Xiang Zeng, Rajive Bagrodia, Mario Gerla GloMoSim: a Library for Parallel Simulation of Large-scale Wireless Networks"; *Proceedings of the 12th Workshop on*

*Parallel and Distributed Simulations -- PADS '98*,  
May 26-29, 1998 in Banff, Alberta, Canada.

Bernard P. Zeigler, Tag Gon Kim, Herbert Praehofer, Theory of Modeling and Simulation, 2nd Edition, Elsevier, January, 2000.

J. Zhou, Z. Ji, M. Takai, and R. Bagrodia. MAYA: Integrating hybrid network modeling to the physical world. ACM Transactions on Modeling and Computer Simulation, 14(2), April 2004.

## **8 AUTHOR BIOGRAPHIES**

**RAJIVE BAGRODIA** is a Professor of Computer Science at UCLA and Founder of Scalable Network Technologies, Inc. He received his B. Tech. in Electrical Engineering from the Indian Institute of Technology and his PhD in Computer Science from the University of Texas at Austin. Dr. Bagrodia's research interests include wireless network design and analysis, mobile systems, and parallel simulation; he has published almost 150 research papers in these and related areas.

**KEN TANG** is the Director of Engineering Services at SNT. He received his B.S. in Information and Computer Science from the University of California, Irvine (UCI), and M.S./Ph.D. in Computer Science from the University of California, Los Angeles (UCLA). His research interests are in the field of ad hoc networking, specifically focusing in the area of ad hoc medium access control (MAC) protocols, reliable multicast and quality of service techniques in tactical packet radio networks.

**STEVE GOLDMAN** is a Boeing Director currently responsible for the development and support of the System of Systems Modeling and Simulation Environment on the FCS Program. He received his BS Electrical Engineering from Rutgers Engineering, BA in Physiology from Rutgers University and MS in Electrical Engineering from Stevens Institute of Technology.

**DILIP KUMAR** is a Sr Manager in the Boeing Company working on the Future Combat System (FCS) program. His current assignment consists of managing a group that deals with Systems Engineering for System of Systems Modeling and Simulation of FCS. Dr. Kumar received his B.Tech from Indian Institute of Technology, Madras and Ph.D. in Aerospace Engineering from Mississippi State University. Dr. Kumar worked on Modeling and Simulation of Boeing Commercial Airplanes Flight Management Systems for over 20 years and moved to Modeling and Simulation of Army Systems 3 years ago with special emphasis on Modeling Communication Effects of large Mobile, Adhoc Wireless Networks.