Design and development of a simulation environment in OPNET using High Performance Computing

Mohammad Z. Ahmad, Mustafa I. Akbas, Damla Turgut
School of EECS
University of Central Florida
Email: {zubair, miakbas, turgut}@eecs.ucf.edu

Brian Goldiez, Ravi Palaniappan
Institute for Simulation and Training
University of Central Florida
Email: {bgoldiez, rpalania}@ist.ucf.edu

Troy Dere
RDECOM-STTC
Email: Troy.Dere@us.army.mil

Keywords:
OPNET, WARSIM, Constructive simulation

ABSTRACT: *Most major simulation federations are designed and constructed based on the professional experience of one or a few experts with extensive experience in creating such federations. Organizations have a risk of losing their understanding of federation performance if one of their key people is no longer available. As very little objective data on federation performance exists, it is difficult for many facilities to create large federations in a manner that insures optimum performance. A detailed architecture and accompanying design guidelines or tools able to create viable interoperable implementations has not yet emerged which minimizes the amount of human intervention, tweaking, and custom software development. This paper proposes the creation of a simulation environment and the generation of initial simulations where data can be gathered that depicts some of the trade-offs needed to make intelligent decisions regarding the design and deployment of simulation federations. The project applies the expertise of University of Central Florida (UCF) in simulation, computer network modeling and high performance computing. The test-bed and data generation/analysis activity was performed in collaboration with Team Orlando, a partnership among the military services, industry, and academia working to leverage resources and contribute to the overall security of the United States. We captured objective data on the performance of a number of very complex federations and created detailed OPNET models of the computers, networks, and software applications that make up large federations. As the communication models were complex and a large number of different federation configurations were needed to be studied, the simulations were run on the new high performance computer (HPC) facility at the UCF. The results of this study will allow the simulation community to design more efficient federations.*

1. Introduction

There has been considerable activity in developing standards and demonstrating instances of connected simulators/simulations since interoperable simulation was first demonstrated in 1992. Additional functionality has been desired to allow interoperation between live, virtual, and constructive simulations and simulators. Success has been achieved on a case by case basis, but what has not yet emerged is a detailed architecture and accompanying design guidelines or tools that are able to create viable interoperable implementations while minimizing the amount of human intervention, tweaking, and custom software development. The Defense Department has commissioned a group of individuals to create an architecture for LVC simulation, known as the Live, Virtual, Constructive Architecture Roadmap (LVCAR) Study [1]. The LVCAR and other groups working in the area of simulator interoperability have a difficult task because little data is accumulated, published, and vetted about what works and what doesn't. Their task is further complicated by the rapid advances in technology and development methods, a dispersed base of users and developers, and the need to quickly get the working prototypes to users.

This paper reports on the creation of a simulation environment and the generation of initial simulations depicting some of the trade-offs needed to make intelligent architectural and design decisions for interoperable simulation systems. As part of the effort described in this paper, UCF has created the necessary infrastructure and expertise to simulate interactions between simulators using this simulation environment on an HPC (High Performance Computer [2]) where repeated runs can be made and analyzed.

The main goal of this study is the creation of simulation models of various federations in order to characterize the network behavior of the federation and analyze data traffic generated and received by federates and the Runtime Infrastructure (RTI). Real-life component models are used for different network configurations and the WARSIM architecture is then modeled. The primary contributions of this work to date are:

- Interpretation, design and partial implementation of WARSIM architecture in OPNET [3]
- Parsing of available WARSIM exercise data to gain better understanding of network traffic characteristics.

The paper is organized as follows. Section 2 presents characteristics of the WARSIM architecture modeled and details the various parsing methodologies adopted on the available data. The development and implementation of the WARSIM architecture in OPNET is given in Section 3 and we conclude in Section 4.

2. Model Selection

In order to create a reliable simulation environment, an existing federation model for development was required.

The first trial for this purpose was a visit to US Joint Forces Command's Development and Integration Facility (JDIF) in Orlando, Florida. A simple Joint Theater Level Simulation/Joint Conflict and Tactical Simulation (JTLS/JCATS) interaction was observed, however no data was obtained from the JDIF visit due to technical problems encountered.

Subsequently, data was requested from MITRE through Army PEO-STRI for a Verification Event/Operational Readiness Event (VE/ORE) testing exercise which used the WARSIM model. The WARSIM model was preferred to the JTLS/JACATS architecture since WARSIM contained more real life scenarios. Approximately 1GB of WARSIM exercise data was provided to UCF by PEO-STRI in mid-August, 2008.

Through interactions with Subject Matter Experts (SMEs), we were able to gain a better insight of the model details. The resulting OPNET model has been developed in accordance with WARSIM architecture.

2.1 WARSIM Architecture

We discuss the highlights gleaned from the WARSIM architecture provided to UCF as follows:

There are total of 11 federates in the WARSIM federation that interact through the RTI. 4 of the 11 federates generate 90% of the data. These 4 federates represent the ground federate models.

Figure 1 shows a representation of a WARSIM model we have used for developing our simulation models. The federates are connected through the RTI based on the High Level Architecture (HLA) and the communication between the different federates is managed by the RTI. The data is filtered from RTI gateway to Battle Station Work Station (BSWS) which receives the minimal data for display on the GUI. The remote gateway distributes the data to other networks.

The WARSIM exercise data provided to us was generated by the data logger and consisted of time stamped data logs of interactions between the various objects and the attribute updates in the federation. The data source and destination federate for each interaction and update along with the data packet size was also logged. Interaction messages are sent across the federation denoting a particular event during the course of the simulation exercise. Object attribute updates are carried out to specifically change one or multiple object attributes due to a particular interaction which has already been sent out. As a result, object update messages are greater in number in comparison to interaction messages.

2.2 Data parsing

Based on the discussions with WARSIM experts and the WARSIM data discussed above, an initial model was built using the OPNET simulator. This model is intended to represent the WARSIM architecture, federates and the RTI interactions. The model was populated by the data (1GB) obtained from the VE/ORE. Interaction with WARSIM SMEs gave us an understanding of the exercise data for parsing. After an analysis of the object, interaction and summary files, the data was mapped to the originating and destination federates. Some of the important aspects about the dataset identified were: all data messages are broadcast and the receiving federate filters the messages that it needs (receiver-side filtering), the communication is always reliable (not best-effort) and the communication cost is based on the number of messages sent.

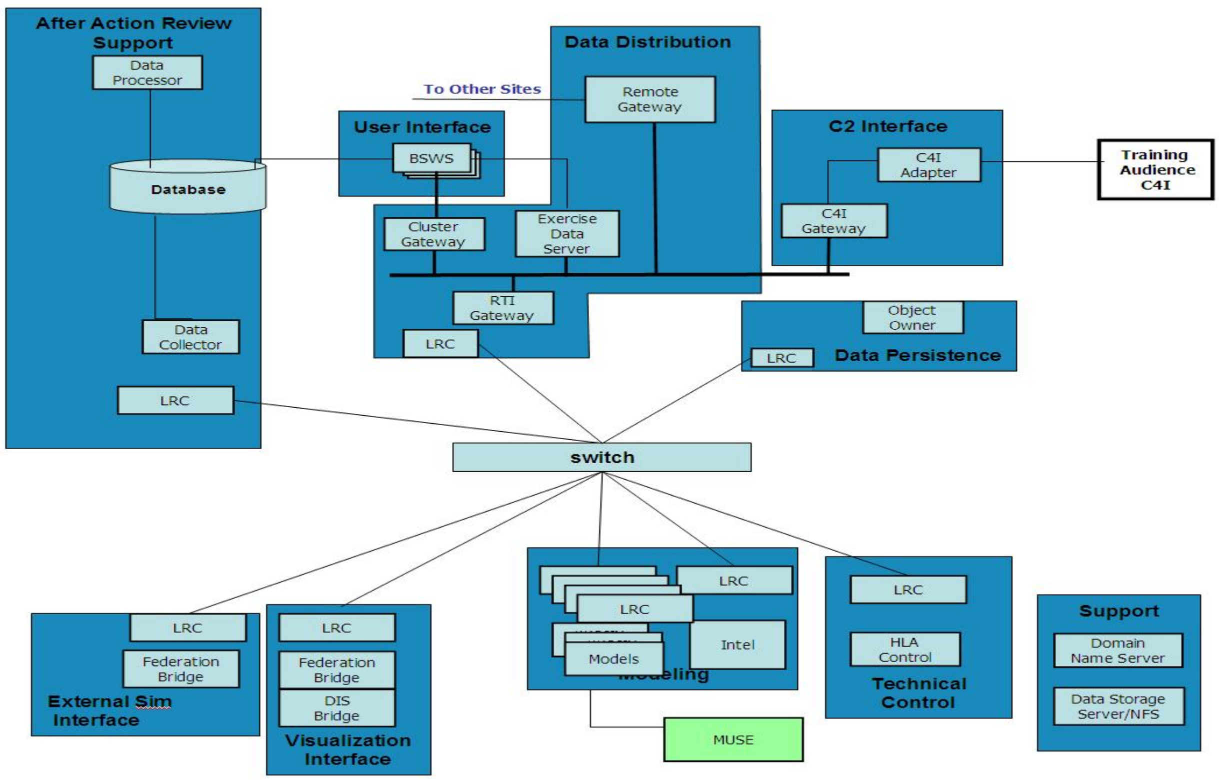


Figure 1. WARSIM Architecture (adapted from WARSIM model provided by MITRE Corp.)

Based on the data analysis, different data components needed to develop a simulation model are extracted. The data sets extracted from the different files are correlated simulation time and include the total number of objects, total number of interactions from each object, number of updates from each object, different types of interactions and the size of each data message.

The data of the WARSIM simulation runs is parsed using simple open source tools GNU-awk (Gawk) and the C programming language. Gawk enables reading data line by line and it has a simple interface to separate comma and/or tab separated values from the simulation log files. It works easily with Linux bash shell scripting, through which command line arguments are provided to the parsing script.

Algorithm 1 represents the pseudo code for determining the total traffic generated due to interactions on the RTI at every time instant. For every interaction type, the interactions file is parsed and the net load size is updated. Since multiple interactions may be sent at the same time, if the time read from the file is different from the last read time, the total load at that time is printed out and the current load is reset back to zero. If the time read is the same as the previously read time, then the current load value is increased by the size of the interaction.

Algorithm 1 Calculating total loads at every time instant

```
while(read individual entry from blobFile)) {
    while(from every entry in interactions file) {
        read time and interaction type;
        if (time does not equal current time) {
           if (current load does not equal 0.0) {
              Output total load at current time
               reset current load counter
           }
         }
        if (interaction type from blobFile equals
interaction type) {
            if (time read equals current time)
              increase current load by interaction size
        }
        set current time to read time
    }
      seek to start of file
}
```

Using Gawk, the unique object types were parsed by checking each line of the data file. For example, in the following line:

```
numberofUpdatesFromObject-LS0130555058=6
```

Extracting text from the delimiters '-' and '=' we conveniently obtain the number of updates from the object LS0130555058, which is 6.

2.3 Results obtained from VE/ORE data

Table 1 below shows the resulting number of data objects, interactions and updates that are being sent out during the entire data logged interval from the VE/ORE simulation exercise for the duration of 54 hours.

Figure 2 shows a time history of the aggregate traffic load during the duration of the exercise event.

Total number of objects found	**36695**
Total number of updates obtained across simulation	**2656337**
Total unique interactions found	**267**
Total number of objects found sending interactions	**520**
Total number of interactions being sent out across simulation	**196872**

Table 1. Source & destination models in VE/ORE data

2.4 Differentiating between source and destination federates

The interactions file provides information about the source and destination federates the interactions are sent across. A federate ID is a specific type with the first four characters representing the model to which the federate belongs. As an example federate ID LS0424977830 denotes that the federate belongs to federate LS04. Table 2 shows the distinct federate IDs that we obtained from the data parsing. The federate IDs starting with the letters "LS" denote the ground models in the architecture. While some federates act only as source (e.g., TC20) or destination (e.g., PS00, PS10), some others act as both (e.g., GS01, LS01, LS02, LS03, LS04).

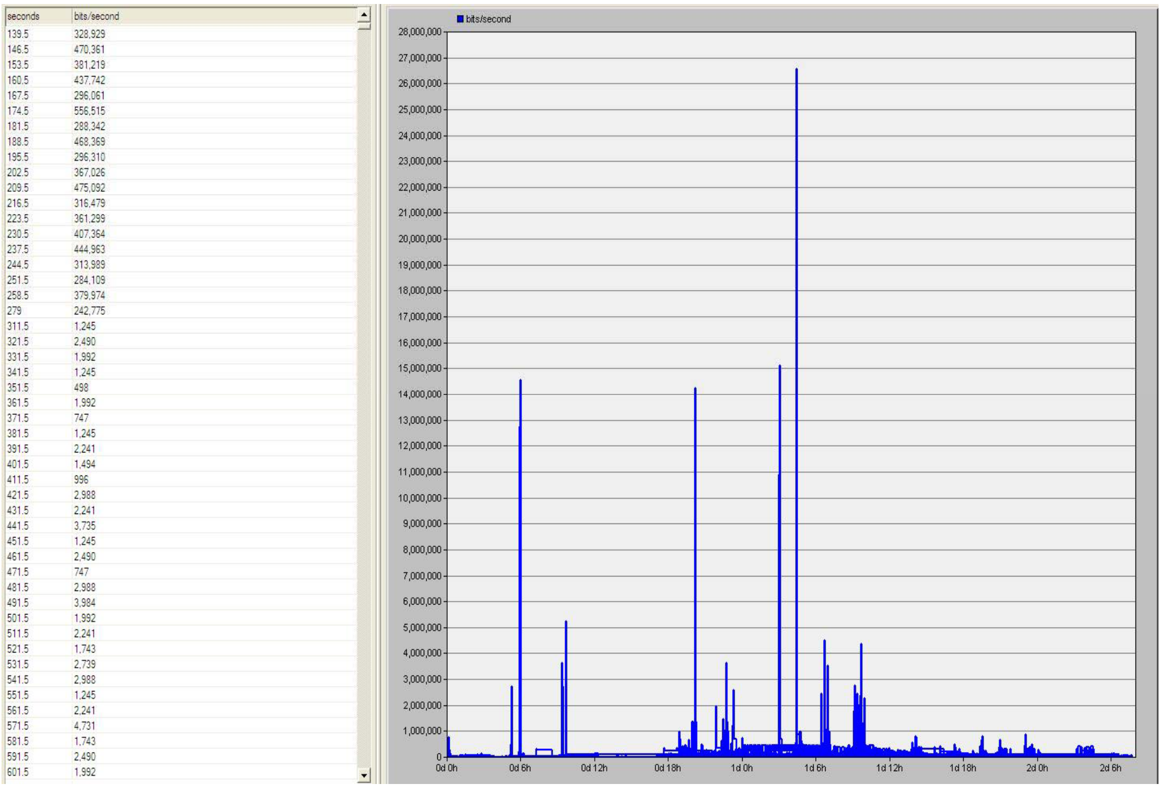
	Federates
Source federates	DUMM, GS01, IS01, JS01, LS01, LS02, LS03, LS04, TC20, TCs0, NULL
Destination federates	1/A/, FAFA, GS01, LS01, LS02, LS03, LS04, PS00, PS10, TAC, UNKN, NULL

Table 2. Federate IDs in VE/ORE data

Once every federate that generates or receives data has been identified we parsed the data to obtain individual interaction traffic loads from every federate to the other federates within the network. Algorithm 2 presents the pseudo code for the algorithm used to carry out this computation. For every source model, we read every destination model ID and then parse through the interactions file for this source destination pair. For every such pair, traffic loads are calculated similar to the previous algorithm for every time instant.

Algorithm 2 Calculating total loads for every source destination pair at each time instant

```
while (read individual entry from srcModel file) {
   while (read individual entry from destModel file) {
          Initialize load = 0
          while (read each timestamped line from interactions file) {
                Extract first four characters from src and dest fields;
                 Compare with variables read from files
                 If (comparison successful) {
                       increase load by the size of interaction
                       Compare current_time with last read time
                       If (times are different) {
                              Output loads at that time
                              Initialize current time with last read time
                              Initialize load = 0
                       }
                 }
          }
}
```



seconds	bits/second
139.5	328,929
146.5	470,361
153.5	381,219
160.5	437,742
167.5	296,061
174.5	556,515
181.5	288,342
188.5	468,369
195.5	296,310
202.5	367,026
209.5	475,092
216.5	316,479
223.5	361,299
230.5	407,364
237.5	444,963
244.5	313,989
251.5	284,109
258.5	379,974
279	242,775
311.5	1,245
321.5	2,490
331.5	1,992
341.5	1,245
351.5	498
361.5	1,992
371.5	747
381.5	1,245
391.5	2,241
401.5	1,494
411.5	996
421.5	2,988
431.5	2,241
441.5	3,735
451.5	1,245
461.5	2,490
471.5	747
481.5	2,988
491.5	3,984
501.5	1,992
511.5	2,241
521.5	1,743
531.5	2,739
541.5	2,988
551.5	1,245
561.5	2,241
571.5	4,731
581.5	1,743
591.5	2,490
601.5	1,992

Figure 2. Traffic load vs. Time obtained from VE/ORE data

Individual traffic loads from each federate to the other helps gain a better understanding of the network traffic load throughout the course of the exercise. The load data obtained from this step can now be generalized to obtain specific statistical traffic distributions.

3. Modeling WARSIM in OPNET

3.1 Development environment

Network simulator programs have grown in maturity since they first appeared as performance, management and prediction tools. There are a considerable number of simulation tools in the market. The main characteristics to classify them are accuracy, speed and ease of use.

There are too many different parameter variations and different possible network scenarios to adequately determine the best simulator. The suitability of the simulators can be validated for the particular case of a project. There were two main simulators selected in the beginning of the project: NS-2 and OPNET. Both of these simulators are very-well known and widely used in academia and industry.

We used the project editor of the OPNET simulator to build a star topology of a small network in an office building. We were able to choose statistics to collect such as delay, utilization and load on the links, and analyze the results. In addition, we tested multiple star topology networks to ensure that the added load by the additional networks should not cause network failure. This example project has similarities with the WARSIM model. The similarities included the network components and network traffic.

The same project was also developed in NS-2. While the statistics and technical results were similar in both simulators, we chose OPNET due to its ease of use, friendly user interface, and the embedded tools for creating and testing networks in a timely manner. Even though NS-2 simulator has a good set of end-to-end network protocols, the OPNET simulator has a large database of network equipment models, including routers and switches from several network equipment vendors currently in use. Therefore, OPNET was chosen as the development environment for the simulation model.

3.2 **OPNET Model**

The main goal when modeling WARSIM is creation of a simulation environment for various federation configurations. In order to simulate the network behavior of the federation, we need to have a model for each federate and the data traffic generated and received by the individual federates and the RTI. To the best of our knowledge, there has been no readily available simulation model for federates and networks of federates (i.e., federations).

The federate models in this study are iteratively created with increasing accuracy based on the interviews with the SMEs. Our goal is not packet level accuracy; rather the level of detail which is sufficient to allow for correct decisions regarding the network topology and simulation setup. However the created simulation environment also provides the ability to make design choices at the packet level if needed. OPNET Simulator offers a Packet Editor to edit the content of the packets.

The high level view of the OPNET model created is shown in **Error! Reference source not found.** The OPNET simulation models are created for the federates in the OPNET simulator and these models have the ability to create and receive simulated traffic. The network components comprising the federates are represented using hardware models from the OPNET library.

The interaction among federates is shown in Figure 3. The dashed line shows the data flow, which is generated in the ground models and transmitted to the Data persistence federate. Transport Control Protocol (TCP) is used as the traffic type since the simulated traffic is reliable.

Figure 4 shows the ground models created in OPNET Simulator. Each Ground Model in this figure has a group of workstations to create or receive traffic and a server for creating server-client applications.

The OPNET architecture is designed with different levels of user selectability. The characteristics of individual components and links can be easily modified as per user requirements. For instance, although the chosen link type for all the links throughout the network was 10 Mbps Ethernet; we can vary the bandwidth of each link as defined by the user. This is an excellent feature that OPNET offers when designing the network models.
Figure 5 shows the user friendly node selection window of OPNET, which offers a very wide range of node types.

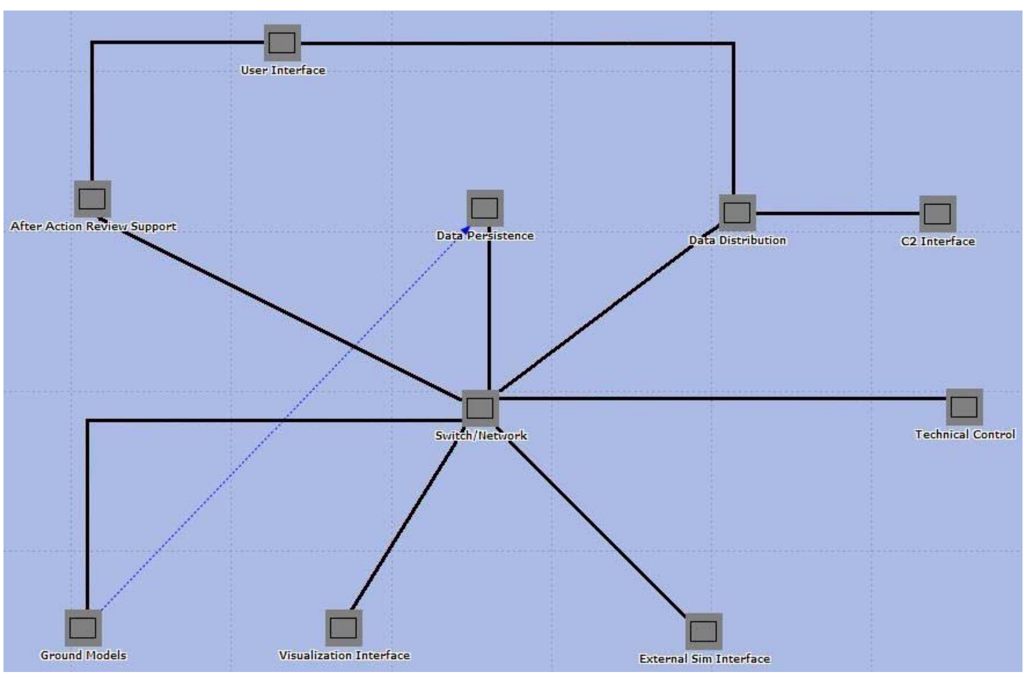


Figure 3. OPNET model based on WARSIM architecture

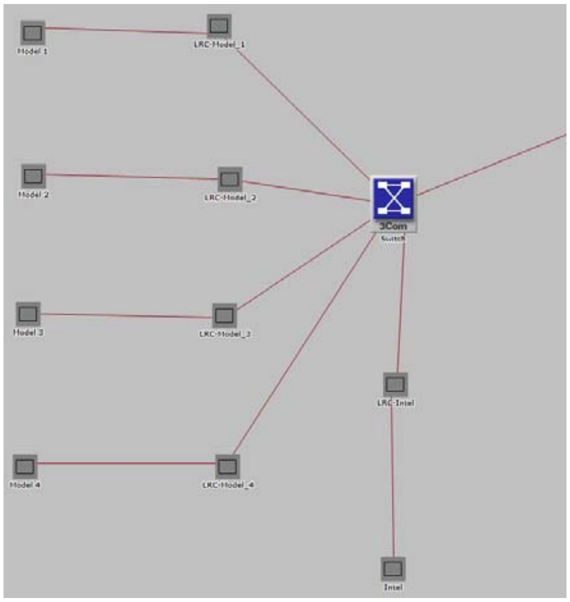


Figure 4. Ground Models created in OPNET Simulator

Initial OPNET network simulations were executed on stand-alone workstations. The simulations were conducted on the UCF High Performance Computing System which greatly improved runtime efficiency. Therefore the number of runs that could be performed was also improved.

The Institute for Simulation and Training at the University of Central Florida operates a high performance computing cluster, called STOKES (see

Figure 6) which consists of over 600 core processors with over 40 TB of storage, high speed inter-processor interconnects using Infiniband and the capability to perform 6.6 trillion floating point operations per second (TFLOPS). The Infiniband interconnects are capable of data transfers to a speed of 40 Giga bits/sec.

3.2.1 Design Considerations

Since there was no information about the individual components of the network supporting the federation, there are several design assumptions needed in creating the OPNET model.

The first design assumption was the need to assign unknown federate ID's to the federates in the OPNET model. The known federate IDs, including LS01, LS02, LS03, and LS04, are assigned to ground models. The unknown federate ID's are assigned to federates randomly. Our assumption was to distribute the traffic throughout the network and create a realistic architecture.

Due to lack of sufficient information on some of the data traffic we did not include them in the simulation network model. For example, even though traffic originating from a "NULL" source was a large portion in the total traffic load it was not modeled in the OPNET simulator. We expect that further meetings with SMEs we will be able to improve and refine these uncertainties in simulation in future iterations of the OPNET simulations and data parsing currently being formulated.

Another design assumption used in simulating the network was that the non-HLA traffic was at a Constant Bit Rate (CBR) and does not flow through RTI switch\network. CBR specifies a fixed bit rate so that data is sent in a steady stream. SME feedback suggested that non-HLA CBR data also flows through the RTI and subsequently we revised our model to include this feature.

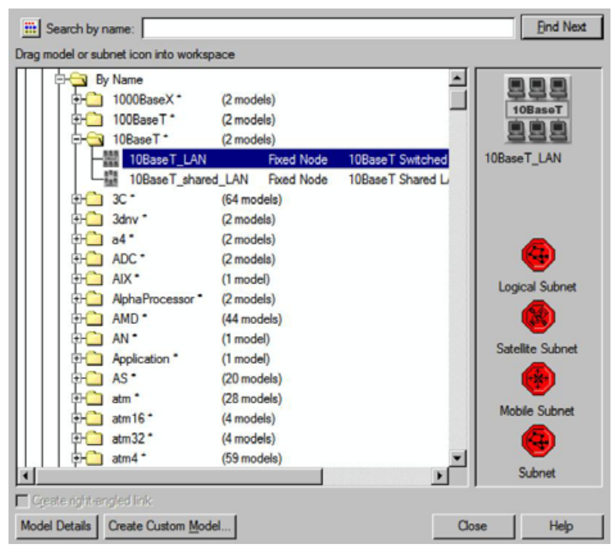


Figure 5. OPNET Node Selection Menu

3.2.2 OPNET Model Characteristics

One of the main characteristics of the OPNET model is the capability to select levels for data flow. The traffic in the network can be defined either as a flow from an individual workstation in a federate to another or as a flow from a group of workstations in a federate to another group. These traffic flows can also be aggregated to simulate the flow between federates. Depending on the needs and goals of the analysis, this selection can be varied throughout the model.

The OPNET model is designed to support reliable (WARSIM) communication. All the traffic created in the network is transmitted using TCP. However the architecture is also capable for best-effort communication and supports multiple network paths. The OPNET model of the WARSIM federation has the ability to measure network traffic load at any point on the network using packet sniffers. The OPNET model has the ability to use any type of traffic in the federation to stimulate the network. The actual data that was obtained from VE/ORE testing exercise is used in the model to demonstrate this ability as a proof-of-concept. Any probability distribution function can also be usedas a substitute for traffic data.

OPNET offers built in models for some COTS equipment. After the first few iterations of the WARSIM architecture were designed, some of these equipment models replaced the sample network components in the network. For instance, Dell Precision workstations replaced the OPNET Ethernet Workstation Node (LAN). These equipments used are the standard templates in the OPNET libraries. However, the user can also change the characteristics of these components to accommodate specific needs. For example, the model of a specific COTS router can be modified by adding one or more Ethernet ports for analysis.

The equipment selection is based on the feedback from the subject matter experts. The default equipments used in creating the OPNET model are:

- Dell Precision Workstation
- 3Com SSII Switch
- Cisco 2514 Series Router
- 10 Base T links

4. Conclusion

There is still a need for a simulation tool where various large federations can be easily simulated to support studying the complex relationship between federates while maintaining optimum performance. In this paper, we have described the initial prototype of a simulation environment and the generation of initial simulations where data can be gathered that depicts some of the trade-offs needed to make intelligent decisions regarding the design and deployment of simulation federations. We have chosen the WARSIM architecture to model in OPNET simulator by feeding the parsed WARSIM exercise data into OPNET.

Figure 6. UCF High Performance Computing Cluster STOKES

5. References

[1] www.discussions.sisostds.org
[2] Stokes HPC at UCF. http://www.ist.ucf.edu/hpc/main_page.html
[3] www.opnet.com

Acknowledgements

This work is funded in part by the Joint Forces Command, JTIEC through the US Army RDECOM W91CRB-08-D-0015. The work contained herein represents the views of the authors and should not be construed as an official position of any of the sponsoring agencies. The authors acknowledge with thanks the support of OPNET Technologies Inc. in using OPNET.

The authors would also like to acknowledge the help of Dr. Roger Smith, Sterling Hall, Richard Crutchfield, Anita Zabek, Mike Wright, Karen Williams, Sergio Tafur, Ola Batarseh and Ed Carmona.

Author Biographies

MOHAMMAD Z. AHMAD is pursuing his PhD in Computer Engineering at University of Central Florida (UCF). He received his BS degree in Information Science and Engineering at VTU in July 2005 and MS degree in Computer Engineering at UCF in 2007. His research interests include wireless sensor and ad hoc networks.

MUSTAFA I. AKBAS is pursuing his PhD in the Computer Engineering at University of Central Florida. He received his BS and MS degrees from the Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey. His research interests include wireless sensor networks.

DAMLA TURGUT is an Associate Professor with the School of Electrical Engineering and Computer Science at the University of Central Florida. She received her BS, MS, and PhD degrees from the Computer Science and Engineering Department of University of Texas at Arlington in 1994, 1996, and 2002 respectively. She has been included in the WHO's WHO among students in American Universities and Colleges in 2002. She has been awarded outstanding research award and has been recipient of the Texas Telecommunication Engineering Consortium (TxTEC) fellowship. She is a member of IEEE, member of the ACM, and the Upsilon Pi Epsilon honorary society. Her research interests include wireless networking, ubiquitous and mobile computing, distributed systems, embodied agents and teamwork collaboration in agents.

BRIAN GOLDIEZ has over 30 years of modeling and simulation experience in industry, government, and academia. As Deputy Director, Dr. Goldiez is involved in strategic planning, organizational matters, outreach, and organizing large, multi-discipline research programs. As Principal Investigator, Dr. Goldiez has been involved in various initiatives including, human and systems performance measurement, computer graphics, interoperability, data base research, and augmented and virtual reality systems. Dr. Goldiez publishes extensively in his areas of research and consults to industry and government (US and foreign) in his areas of expertise. He also teaches and advises modeling and simulation graduate students. Goldiez has a BS in Aerospace Engineering, an MS in Computer Engineering, and was UCF's first recipient of a Ph.D. in Modeling and Simulation. He is also a Certified Modeling and Simulation Professional (Charter Member) and Senior Member of the IEEE.

RAVI PALANIAPPAN is a Research Associate at the Institute for Simulation & Training, University of Central Florida. He received his MS degree in Electrical Engineering from UCF and currently pursuing his PhD in Modeling and Simulation. His areas of interest include wireless communication and High Performance Computing.

TROY DERE is a Program Manager at the Research Development and Engineering Command - SFC Paul Ray Smith Simulation & Technology Training Center (RDECOM – STTC). His areas of interest include High Performance Computing and LVC Simulation.