Modeling a University Computer Laboratory using OPNET Software

Vasil Hnatyshin*, Andrea F. Lobo*, Pavel Bashkirtsev,
Robert DeDomenico, Andrew Fabian, Gregg Gramatges,
James Metting, Mike Simmons, and Matt Stiefel

{bashki12, dedome96, fabian78, gramat64, mettin61,
simmon51, stiefe93}@students.rowan.edu
*{hnatyshin, lobo}@rowan.edu

Computer Science Department
Rowan University
201 Mullica Hill Rd
Glassboro, NJ 08028

Abstract

Networking hardware and software can be both complicated and expensive. Companies wish to better understand the anticipated benefits of new networking resources before making additional purchases. In such situations, simulation and modeling is a quick and inexpensive way of studying multiple scenarios and identifying the best possible configuration. The goal of this project is to examine student and faculty usage of network applications and its effects on the Rowan University network. This paper presents a simulation model of a Computer Science Department's undergraduate laboratory at Rowan University. We developed application and user profiles to model the applications used by the students in this laboratory. These applications include FTP clients, e-mail transfer agents, web browsers, instant messengers, and other applications that rely on the campus network and the Internet for data transfer. We created a custom built OPNET model of the AOL Instant Messaging application. Next we built a model of the laboratory's physical layout and applied the developed application and user profiles to the workstations in the layout. We compared results obtained via simulation with live packet traces collected in the laboratory, and adjusted the simulation configuration accordingly. The simulation was developed using the OPNET Modeler[2] software package, and the packet traces were collected and analyzed using Ethereal[1], a public-domain protocol analysis software.

1. Introduction

Networking hardware can be both complicated and expensive. In today's competitive world, companies wish to better understand the anticipated benefits of new networking resources before making additional investments. In such situations, simulation and modeling is a quick and inexpensive way of studying multiple scenarios and identifying the best possible configuration. For example, an expanding company that plans to add a set of new offices may want to research options for upgrading their network infrastructure. Specifically, the company may want to know if existing network capacity is sufficient to carry additional traffic generated by the new offices, how the projected network resource consumption will change, and what the best options are for upgrading the network. Often the quickest and the most inexpensive way to get the answer to these questions is to create a model of the existing network and study possible scenarios of the network expansion via software simulation. Simulation is an excellent tool for studying performance and identifying the cause of problems in the network.

In this paper we discuss our ongoing efforts to develop a model of the Rowan University network and to study its performance using the OPNET Modeler [2] network simulation software. The goal of this study is to develop a comprehensive model of the Rowan University's wired and wireless networks which will include:

- open student computer laboratories,
- faculty and administration offices,
- networked computer equipment for conducting research,
- the campus backbone, and
- the university's Internet Service Provider.

This paper describes the first step in this study: Creating a simulation model of an undergraduate student computer laboratory which includes modeling its network topology, applications, and user profiles. In addition, this paper describes the simulation model of AOL's Instant Messenger software developed using OPNET Modeler [2].

2. Methodology

To develop a simulation model of the student computer laboratory, we examined and studied the physical topology of the local area network in the laboratory, the application usage profiles of the lab users, and the applications that require network access.

To verify the correctness of the developed simulation models, we planned to compare simulation results with the live packet traces obtained from a Rowan University Computer Science Department laboratory, located in 303 Robinson Hall. To simplify the verification process, we first created a small network model to examine and validate the implementation of individual application models. We compared simulation results with live packet traces obtained from the network. The simulation models of individual applications were developed using the OPNET Modeler [2] network simulation software. The packet traces from the network were collected and analyzed using Ethereal [1], a public-domain protocol analysis software. Once all application models were verified, we deployed them into the simulation model of the laboratory.

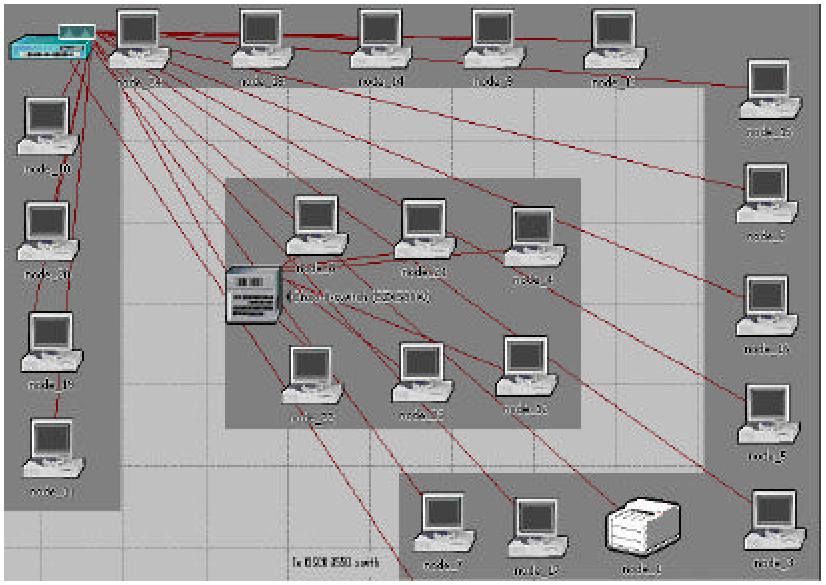


Figure 1. Topology of Robinson 303 student laboratory

2.1. Network Topology

We examined the connectivity and hardware configuration of the computers in the Robinson 303 laboratory. Figure 1 illustrates the physical topology of the laboratory. The laboratory consists of twenty-two Dell GX270 workstations and a laser printer connected to a CISCO 303 Catalyst 2950 switch. The printer and eighteen workstations that run the Windows XP operating system are directly connected to the CISCO 303 Catalyst 2950 switch. The other six workstations run FreeBSD and are connected to the CISCO 303 Catalyst 2950 switch via a CISCO 8-port switch.

The Robinson 303 student laboratory model is connected to a simplified model of the Rowan University network and its Internet connection. We modeled the Rowan University network as a CISCO 3550 switch that connects to a cluster of local servers, the Robinson 303 laboratory, and the Rowan University Internet gateway router. Other connections to the switch are ignored in this version of the model but can be easily added later on. The Internet is simulated using an IP32 cloud, a built-in OPNET model. To access application services over the Internet, the traffic that originates from the Robinson 303 laboratory must travel via the Rowan University Internet gateway router, the IP32 cloud, and the Internet Servers gateway. Figure 2 illustrates the topology of the Rowan University network and its Internet connections.

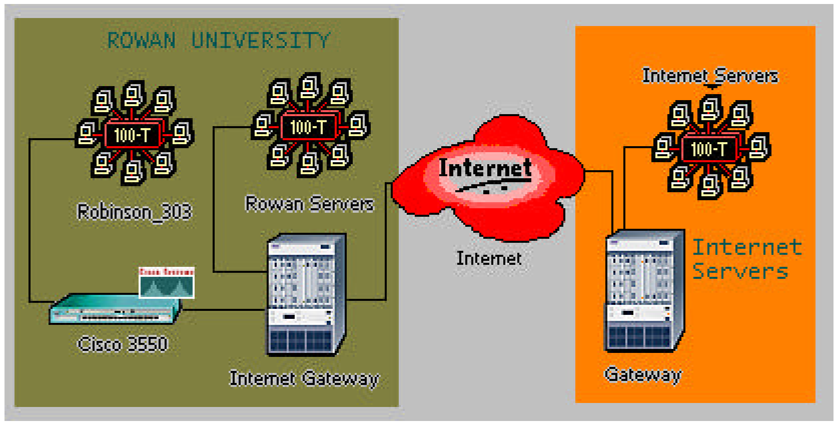


Figure 2. Rowan University Network topology

2.2. Applications

Next we identified the most commonly used applications that require network access. We divided these applications into two categories: (1) those that require Internet access, which we call *the Internet applications* and (2) those that obtain the necessary services on the local Rowan University network, which we call *the local applications*. Internet application traffic must travel via the Rowan University Internet gateway, the IP32 cloud, and finally via the Internet Servers gateway to reach the Internet Servers. The Internet applications are *web browsing*, *instant messaging*, and *online games*.

The local applications access the services they need via the CISCO 3550 switch without ever traversing the Internet. The local applications are *remote login*, *e-mail*, *browsing the Rowan University web pages*, and *FTP*. Students usually use remote login and FTP applications to access local Rowan University servers to complete various programming assignments or save data on the network drives. Students access local e-mail servers to read their daily e-mail. Finally, students access the Rowan University web pages for various school and course-related information. Currently we do not model any communication between the local servers and the Internet because they have very little influence on the traffic generated in the laboratory.

We classified the Internet and local applications described above into two broad categories: *leisure* and *homework*. We further divided each category into additional sub-classes as follows:

- **Leisure:** *gaming* – which is 90% of time playing online games and 10% using the Instant Messenger,
- **Leisure:** *web-browsing* – which is 60% browsing the Internet, 10% accessing e-mail, and 30% using Instant Messenger,
- **Homework:** *writing a term paper* – which is 60% using FTP program to save data on the local network drive, 10% reading e-mail, 25% browsing the Internet to find information on the subject, and 5% browsing Rowan University web pages to retrieve course-related information.
- **Homework:** *programming* – which is 60% using remote login to complete programming assignments, 10% reading e-mail, 25% browsing the Internet to find information on the subject, and 5% browsing Rowan University web pages to retrieve course-related information.

Table 1 summarizes the application classifications. Even though there are other network applications used in the computer laboratory, this study concentrates on the applications most frequently used in the Robinson 303 laboratory. The model can be easily expanded to include additional applications.

Application Category	Sub-category	Local Applications	Internet Applications
Leisure	*Web-browsing*	• E-mail (10%)	• Web (60%) • Instant Messenger (30%)
	Online Gaming	• None	• Gaming (90%) • Instant Messenger (10%)
Homework	*Writing a paper*	• FTP (60%) • E-mail (10%) • Web (5%)	• Web (25%)
	Programming	• Remote Login (60%) • E-mail (10%) • Web (5%)	• Web (25%)

Table 1. Application Classification

2.3. User profiles

We developed a set of *user profiles* which specify how the network applications are being used in the computer laboratory. First, we divided the user profiles based on the times during the semesters into the following categories: *exam time*, *weekdays*, and *weekends*. These categories determine lab occupancy (e.g. the number of simultaneously active users in the laboratory) and the application selection (e.g. which applications are being used).

The *exam time* happens twice during the semester, at midterm and at end of semester, and, during these periods, the lab is completely occupied with students working on their homework and programming assignments only. The *weekdays* time period models student activity on Monday through Friday. During this time period, the laboratory is not completely occupied, and students mostly work on their homework assignments and occasionally run leisure applications. Finally, the *weekends* time period includes Saturday and Sunday, when the laboratory is primarily empty. On weekends students mostly run leisure applications and occasionally work on homework assignments. We further divided each user profile based on the time of the day into four sub-categories: *morning*, *day*, *evening*, or *night.* Table 2 summarizes the user profiles and lists the application distribution and laboratory occupancy.

User Profile	Time of the Day	Lab Occupancy	Application Category
Exam Time	*Morning*	0%	• N/A
	Day	100%	• 100% Homework
	Evening	100%	• 100% Homework
	Night	50%	• 90% Homework • 10% Leisure
Weekdays	*Morning*	20%	• 100% Homework
	Day	60%	• 100% Homework
	Evening	80%	• 80% Homework • 20% Leisure
	Night	20%	• 50% Homework • 50% Leisure
Weekends	*Morning*	0%	• N/A
	Day	20%	• 30% Homework • 70% Leisure
	Evening	10%	• 10% Homework • 90% Leisure
	Night	0%	• N/A

Table 2. User profiles

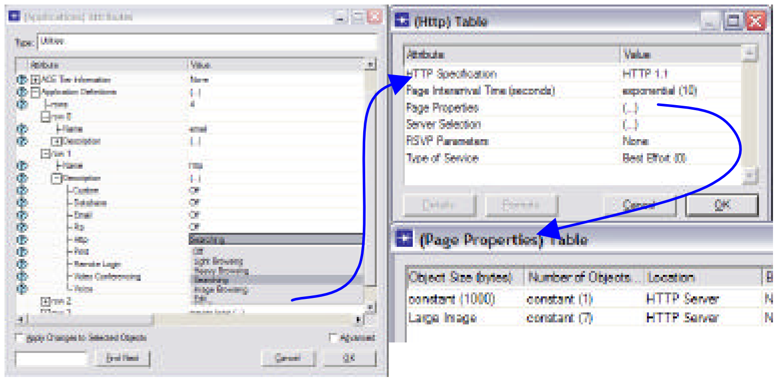
For simplicity, we assumed that, for each application category, students execute the sub-category applications with equal frequency. For example, at night during the exam time period the laboratory is only 50% occupied and out of those users, 90% of users are doing homework assignments; 45% are writing papers and another 45% are working on programming assignments. The other 10% of the users run leisure applications; 5% are playing online games and another 5% are doing leisure web browsing. The simulation model can be easily modified to have sub-category applications used with different frequencies.

3. Modeling Applications in OPNET

We modeled the user applications via OPNET Modeler [2] simulation software. OPNET Modeler provides standard built-in models for software applications such as *web (HTTP)*, *e-mail*, *FTP*, and *remote login*, which can be easily configured to simulate applications used in the computer laboratory. However, there are no built-in models for applications such as the Instant Messenger and online gaming. Thus, these applications have to be implemented and configured via OPNET's Custom Application feature. Sections 3.1 through 3.5 briefly describe the configuration steps for each application.

3.1. Configuring Web (HTTP) applications

To set-up and configure any application in OPNET, the user must add the Application Configuration and Profile Configuration modules. The Application Configuration module contains the application definitions, while the Profile Configuration module contains the profiles of user behavior, e.g. describes how the users employ the applications defined in the Application Configuration module. To configure a web browsing application, the application named **HTTP** should be selected from the list of built-in models. OPNET provides pre-set configurations such as: *Light Browsing*, *Heavy Browsing*, *Searching*, or *Image Browsing*. In addition, OPNET allows configuring web applications via parameters such as: *HTTP Specification* which defines the version of HTTP protocol, *Page Interarrival Time* which specifies time in seconds between consecutive webpage downloads, *Page Properties* which models properties of the webpage by specifying the number and the type of objects. Figure 3 illustrates steps for configuring a web application in OPNET.



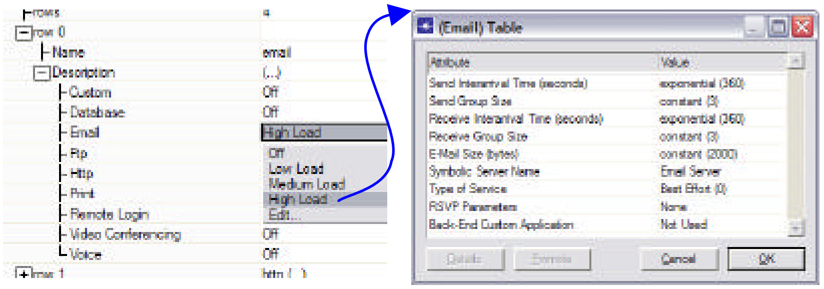
Attribute	Value
HTTP Specification	HTTP 1.1
Page Interarrival Time (seconds)	exponential (10)
Page Properties	(...)
Server Selection	(...)
RSVP Parameters	None
Type of Service	Best Effort (0)

Object Size (bytes)	Number of Objects...	Location
constant (1000)	constant (1)	HTTP Server
Large Image	constant (7)	HTTP Server

Figure 3. Configuring a web application in OPNET

3.2. Configuring e-mail applications

OPNET provides preset configurations for e-mail application as well. The preset E-mail configurations include *Low Load*, *Medium Load*, and *High Load* as shown in Figure 4.



rows	4
row 0	
Name	email
Description	(...)
Custom	Off
Database	Off
Email	High Load
Ftp	
Http	
Print	
Remote Login	
Video Conferencing	Off
Voice	Off
row 1	http (...)

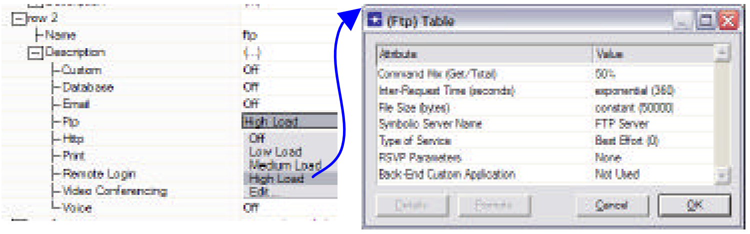
Attribute	Value
Send Interarrival Time (seconds)	exponential (360)
Send Group Size	constant (3)
Receive Interarrival Time (seconds)	exponential (360)
Receive Group Size	constant (3)
E-Mail Size (bytes)	constant (2000)
Symbolic Server Name	Email Server
Type of Service	Best Effort (0)
RSVP Parameters	None
Back-End Custom Application	Not Used

Figure 4. Configuring an e-mail application in OPNET

The model also allows custom configuration via parameters such as *Send(Receive) Interarrival Time* which specify the amount of time in seconds between consecutive sent (receive) operations, *Send(Receive) Group Size* which determine the number of e-mails messages per single sent(receive) operation, and *E-Mail Size* which defines the size of e-mail message in bytes.

3.3. Configuring FTP applications

OPNET also provides preset configurations for FTP application. The preset FTP configurations include *Low Load*, *Medium Load*, and *High Load* as shown in Figure 5.



row 2	
Name	ftp
Description	(...)
Custom	Off
Database	Off
Email	Off
Ftp	High Load
Http	
Print	
Remote Login	
Video Conferencing	
Voice	Off

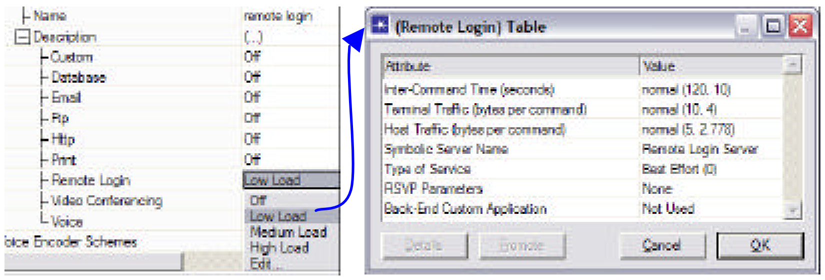
Attribute	Value
Command Mix (Get/Total)	50%
Inter-Request Time (seconds)	exponential (360)
File Size (bytes)	constant (50000)
Symbolic Server Name	FTP Server
Type of Service	Best Effort (0)
RSVP Parameters	None
Back-End Custom Application	Not Used

Figure 5. Configuring an FTP application in OPNET

The model also allows for a custom configuration of FTP applications. The *Command Mix* parameter specifies the ratio between the number of get commands and the total number of FTP requests. For example, when the *Command Mix* parameter is set to its default value of 50% then half of FTP requests will be to get (download) data and the other half to put (upload) data. The *Inter-Request Time* parameter is the time in seconds between consecutive FTP requests. The *File Size* defines the size in bytes of the FTP file to be transferred.

3.4. Configuring Remote Login applications

Similar to E-mail and FTP applications, OPNET provides preset *Low Load*, *Medium Load*, and *High Load* configurations as well as the ability to specify user-defined settings for Remote Login application as shown in Figure 6.



Name	remote login
Description	(...)
Custom	Off
Database	Off
Email	Off
Ftp	Off
Http	Off
Print	Off
Remote Login	Low Load
Video Conferencing	
Voice	
oice Encoder Schemes	

Attribute	Value
Inter-Command Time (seconds)	normal (120, 10)
Terminal Traffic (bytes per command)	normal (10, 4)
Host Traffic (bytes per command)	normal (5, 2.778)
Symbolic Server Name	Remote Login Server
Type of Service	Best Effort (0)
RSVP Parameters	None
Back-End Custom Application	Not Used

Figure 6. Configuring a Remote Login application in OPNET

The *Inter-Command Time* parameter specifies the time in seconds between consecutive commands during the remote login session. The *Terminal(Host) Traffic* parameter defines the size in bytes of each command generated at the terminal (host).

3.5. Configuring AOL Instant Messenger applications

Instant messenger and online gaming are not part of the standard OPNET application models, and they are more complex to simulate. So far we have examined and modeled the Instant Messenger application only. First, we researched the implementation of the AOL Instant Messenger (AIM) software. Then we modeled the Instant Messenger software using OPNET's Task

Configuration module which specifies the packet exchange between the application nodes. Finally, we compared the amount of traffic generated by the Instant Messenger application modeled using OPNET software with live AIM packet traces collected in the network. Based on the comparison, we adjusted the AIM configuration in OPNET until we obtained results that closely resemble the AIM packet trace.

Summary of AIM implementation

The AOL Instant Messenger is implemented as follows. The AIM user must log in to an authorization server with a valid username and password before he/she can send any messages. Upon successful login, the authorization server forwards a cookie to the AIM user. The authentication cookie is required to access the Basic Oscar Service (BOS) server, which manages the message exchange between the AIM users. Oscar is the name of the protocol used for message exchange between instant messenger users. The AIM file transfer is accomplished by creating a direct TCP connection between the AIM users who want to exchange information [3, 4].

Modeling AIM in OPNET

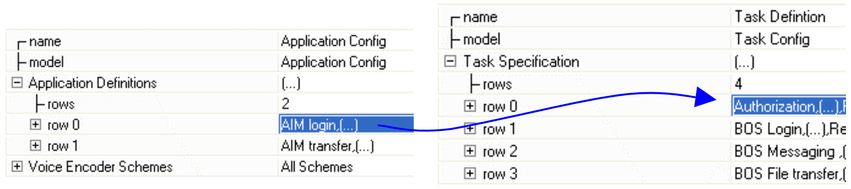
To implement the AIM application, we used the Custom Application and the Task Configuration modules. The Task Configuration module specifies a basic unit of user activity within the custom application, such as a server login or reading an e-mail message [2]. The Custom Application module specifies applications that use the tasks defined in Task Configuration module. We modeled AIM behavior via four basic tasks:

- *Authorization* – log in to authorization server
- *BOS Login* – log in to BOS server
- *BOS Messaging* – message exchange among AIM users via BOS server
- *BOS File transfer* – data exchange among AIM users via direct TCP connection

We used these tasks to implement AIM via two custom applications:

- *AIM Login* – log in to authorization and BOS servers
- *AIM Transfer* – message exchange and file transfer using AIM

The AIM Login is implemented as a one-time sequence of Authorization and BOS Login tasks. AIM Transfer is implemented as a weighted random selection of BOS messaging and BOS file transfer tasks with corresponding weights of 90 and 10, respectively. Figure 7 illustrates the Custom Application and Task Configuration modules set-up.



name	Application Config
model	Application Config
⊟ Application Definitions	(...)
rows	2
⊞ row 0	AIM login,(...)
⊞ row 1	AIM transfer,(...)
⊞ Voice Encoder Schemes	All Schemes

name	Task Defintion
model	Task Config
⊟ Task Specification	(...)
rows	4
⊞ row 0	Authorization,(...),F
⊞ row 1	BOS Login,(...),Re
⊞ row 2	BOS Messaging ,(
⊞ row 3	BOS File transfer,(

Figure 7. Application and Task Configuration

The Authorization and BOS Login tasks are configured as a request-response message exchanges between a single client and the corresponding server. Since all AIM users can talk to each other via the BOS server, we implemented the BOS Messaging task as an exchange of messages between an AIM client and BOS server and between BOS server and another random AIM client. The BOS File transfer is implemented as a direct TCP connection between two randomly selected AIM clients.

4. Validating the AIM model

The biggest challenge in modeling the AIM application was to determine the correct configuration of the parameters that determine the size of each message and the inter-request time between messages in the BOS Messaging task. Other configuration parameters such as the size of the login message and the size of the authorization server response were obtained from [3, 4].

We collected over 10.5 hours of AIM packet traces generated by four different AIM users. We used the Ethereal [1] software to collect data and to filter out AIM traffic. First, we examined collected traces to determine the proper packet size. Each of the traces reported the average packet size to be around 205 bytes with the average packet size per trace ranging between 176 and 240 bytes/packet. Such behavior is best modeled by normal distribution because the packet size appears to have a clearly defined mean of 205 bytes/packet with positive and negative deviations equally likely. We configured the simulation to have packet size normally distributed with a mean outcome of 205 bytes and variance of 20 bytes.

Next, we examined collected traces to determine the inter-request times between consecutive AIM packets. We observed that the inter-request time primarily depends on the frequency of the messages generated by the AIM user. If the user actively communicated with multiple AIM users, then the packet generation frequency varied between 1080 and 1740 packets per hour. We call such behavior *high-intensity user*. On the other hand, if the user only occasionally talks to other AIM members while using the computer to work on some other task, then the average packet generation frequency varies between 102 to 186 packets per hour. We call such behavior *low-intensity user*.

OPNET [5] recommends modeling such user behaviors with the exponential distribution, which assumes that requests are independent of each other. We configured the OPNET simulation to have inter-request times for high- and low-intensity users modeled using exponential distribution with mean outcomes of 2.6 seconds and 40 seconds respectively.

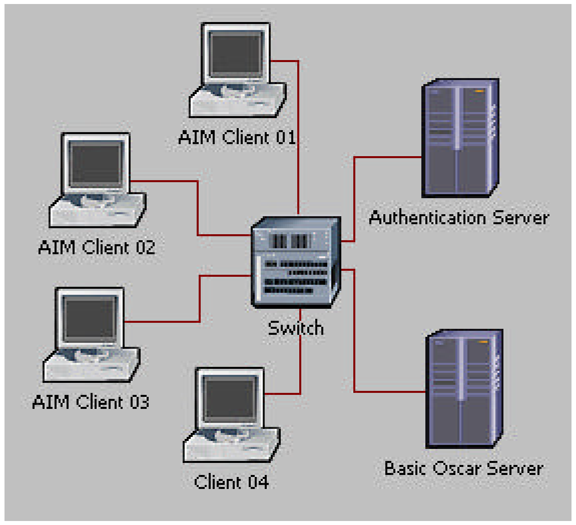


Figure 8. Network topology used to evaluate the AIM implementation

To validate correctness of the Instant Messenger implementation in OPNET we set up a simple OPNET simulation and compared the traffic rate generated by the AIM application in the real network with the data collected from the OPNET simulation. We used a simplified network topology that consists of four workstations connected to authentication and Basic Oscar Service servers via a single router, as illustrated in Figure 8. This simple network topology simplified the debugging process significantly. Table 3 compares the data from the OPNET simulation results and the Ethereal packet traces of the AIM application.

	Low Intensity User		**High Intensity User**	
	Packets/second	Bytes/packet	Packets/second	Bytes/packet
Ethereal trace	**0.0408**	**198**	**0.3016**	**208**
OPNET simulation	**0.0402**	**205**	**0.0311**	**205**

Table 3. Comparison of OPNET and Ethereal AIM traffic

We collected OPNET simulation results for high-intensity and low-intensity users. We ran three 1000-second simulations using randomly selected seed values for each user type. We averaged the data based on the results collected from the individual AIM client workstations.

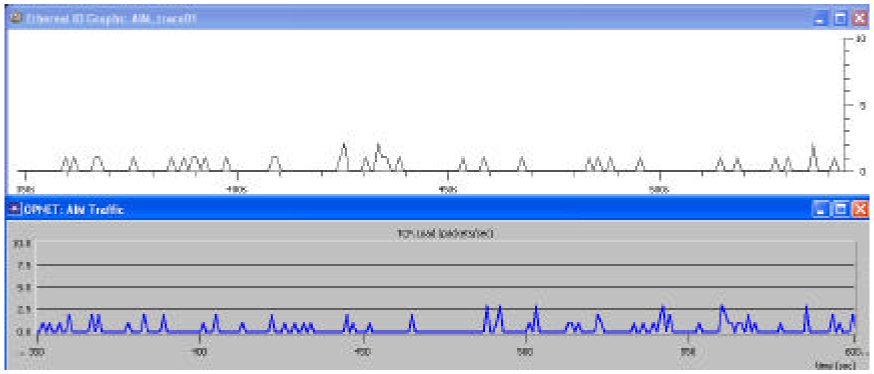


Figure 9. Comparison of TCP traffic generated by the actual and the modeled AIM applications

Figure 9 contains 250 seconds of TCP traffic generated by randomly chosen live and simulated AIM sources. The top of the figure illustrates the Ethereal packet trace of a single AIM client. The bottom of the figure illustrates the traffic obtained from a single AIM client in the OPNET simulation. The behavior is similar.

5. Conclusion

This paper discusses the first steps in ongoing research efforts to model the Rowan University network. In particular, this paper describes the methodology developed for modeling applications and user profiles in a computer laboratory, and explains the steps for modeling a non-standard application such as AOL Instant Messenger using OPNET Modeler. The AIM application model was validated against live traces.

Plans for future work include further refinement of the AIM application model to include periodic BOS server message updates. In addition, the AIM traffic collected through Ethereal had to traverse the Internet to reach the BOS server, while the simulated AIM clients were connected to the BOS server via a single switch. We plan to investigate the influence of network conditions on the accuracy of comparison between the OPNET simulation traffic and the Ethereal trace. We also plan to apply the methodology used to develop the AIM simulation model to create a simulation model of online gaming applications. Finally, we will continue developing a comprehensive model of the Rowan University network, starting from the Computer Science Laboratory in 303 Robinson Hall.

6. References

[1] Ethereal: A Network Protocol Analyzer, http://www.ethereal.com, accessed 8/15/2006.

[2] OPNET Technologies, Inc. http://www.opnet.com/, accessed 8/15/2006.

[3] Gaim 1.5.0: A multi-protocol instant messaging (IM) client, *http://gaim.sourceforge.net/protocol.php*, accessed 8/15/2006.

[4] AIM/Oscar Protocol Specification, *http://www.oilcan.org/oscar/*, accessed 8/15/2006.

[5] OPNET Technologies, Inc. OPNET Modeler Product Documentation, release 11.5.

7. Biographies

This work was conducted by the members of the Networking Club at Rowan University. The Club consists of undergraduate students and faculty members of the Computer Science Department at Rowan University who are interested in data communications and computer networks. The Networking Club was established in 2005 and is supervised by Dr. Hnatyshin and Dr. Lobo. The research interests of the Networking Club include network simulation and modeling, quality of service in IP networks, protocol design, and wireless communications. The Club extensively uses OPNET Modeler and OPNET IT Guru to conduct its research studies.