USING OPNET TO SIMULATE THE COMPUTER SYSTEM THAT GIVES SUPPORT TO AN ON-LINE UNIVERSITY INTRANET

Norbert Martínez[1], Angel A. Juan[2], Joan M. Marquès[3], Javier Faulin[4]

{1, 3, 5} [norbertm@uoc.edu , jmarquesp@uoc.edu]
Computer Science Studies
Open University of Catalonia (Spain)

{2} [angel.alejandro.juan@upc.edu]
Dep. of Applied Mathematics I
Technical University of Catalonia (Spain)

{4} [javier.faulin@unavarra.es]
Dep. of Statistics and Operations Research
Public University of Navarra (Spain)

ABSTRACT

In this paper, we present a preliminary discrete-event simulation study carried out on the computer system that gives support to the intranet of the Open University of Catalonia. The main purpose of this study, developed with the OPNET simulation software, was to help the computer network managers to obtain a better understanding of its internal operation. Other objectives of our paper are related to the discovery of possible performance problems (bottle necks, weak points in the structure, etc.), and to the testing of new designs of the network that could increase its performance, reliability and scalability levels. Finally, we discuss how discrete event simulation software can help computer networking students to improve their practical knowledge on this subject.

Keywords: Simulation, Computer Science Applications, Network Design

1. INTRODUCTION

Computer networks have become extremely important in our day-to-day life, since most companies and institutions depend on the appropriate functioning of their computer networks.

In order to analyze computer networks performance, both analytical and simulation methods can be used. Analytical methods are based upon mathematical analysis that characterizes a network as a set of equations. This approximation usually implies to use several restrictive assumptions, which tend to be not too much realistic, since networks are complex systems formed up by hardware and software (protocols, applications, queueing policies, etc.). On the other hand, simulation techniques can be used to model in detail the dynamic nature of real computer networks **[Law, Kelton; 2000] [Banks, et al; 2001]**. Simulation allows engineers to test different network designs (even before the network physically exists) and to perform what-if analysis with models of the already existent networks without exposing them to failures or inoperative periods.

2. THE CASTELLDEFELS PROJECT

The Open University of Catalonia (UOC) is an on-line university with about 37,000 community members, including students from Spain and Latin America, professors, managers and other staff. With this volume of potential intranet users, performance fine-tuning of the computer system that gives support to the UOC intranet becomes a major task for system managers. For that reason, a group of professors (from areas such as computer networking, operations research and simulation), network managers and computer science students started the so called Castelldefels Project (named behind the city where the computer system is located). The main objective of this project is to improve the system performance levels (and, consequently, to increase the quality of service offered to intranet users) by selecting optimal values for configuration parameters such as network topology, hardware devices, queuing and balancing policies, protocols, etc.) **[Kurose, Ross; 2005] [Peterson, Davie; 2003]**.

Special attention has to be paid to two concrete aspects of the network system: the load balance mechanisms and the session persistence requirements.

The computer system makes use of load balancing mechanisms, which allow a convenient load distribution (requests from distinct users) among different available servers. Two dedicated hardware devices perform this load balancing task. While one of the load balancers is operating, the other is in stand-by status. That way, maintenance tasks can be done without having to stop the web service and, moreover, this service will still being available even if the active load balancer fails. These load balancers are responsible for assuring readiness, at any moment, of web services (HTTP, HTTPS, FTP, SMTP and other proprietary applications). Different load assignment policies can be set up in the balancer. This way, the balancer device decides which available server will attend an incoming user request based on one of the following selection criteria: randomly, sequentially, less loaded server, smallest response-time server, etc...). In order to check if a concrete server is operative, the load balancer carries out regular tests on the servers. This monitoring process can also be configured with different optional parameters (such as elapsed time between two consecutive tets).

For some services, it is also very important to maintain a specific user session in the same front-end (that is to say, to guarantee persistence of the session): when a user introduces its login and password and gets access to the intranet, all information associated to her (profile, mailbox, etc..) is loaded from the database into the front-end. Therefore it is not optimal to reassign her to a different front-end each time she does a new request (since that would add unnecessary load to servers). There are different policies to maintain a user session: using the origin IP address, inserting a cookie in the user PC, etc..).

At the end, quality of service will depend on how fast, reliable and error-free is the service we offer to the intranet users.

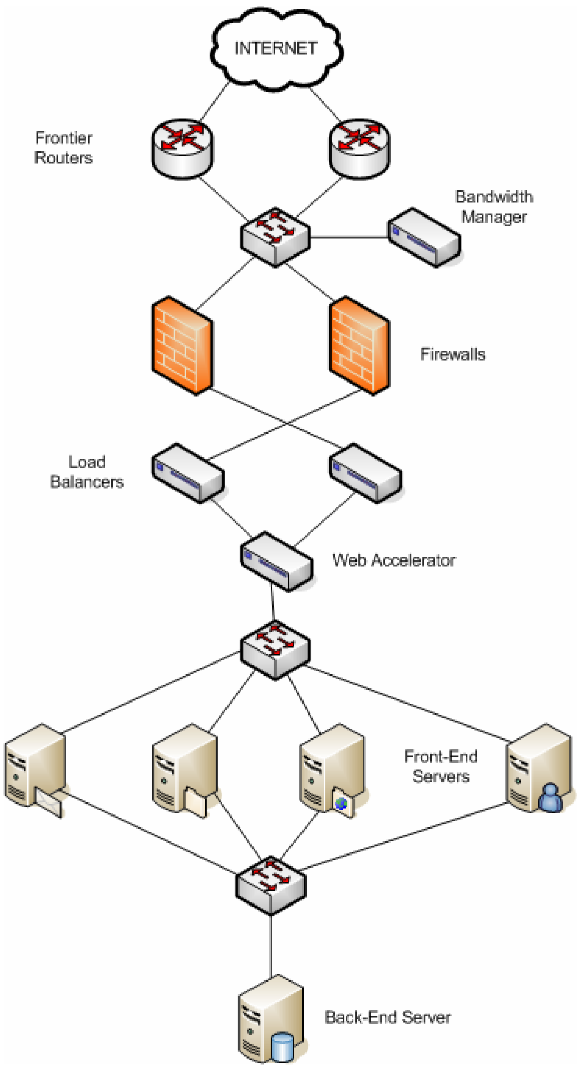

Figure 1: Network system

3. NETWORK SYSTEM DESCRIPTION

Figure 1 shows a graphical representation of the network system under consideration. When a user opens a web browser and types the University URL, a portal page is loaded into the user browser (clients requests are balanced among three portal servers). To complete that process, the client request has already passed through the frontier routers, crossed the firewalls, and arrived to the load balancers, where a new session has been settled down with an available portal server.

Once the user has introduced her login and password (and they have been validated in the database), she enters the intranet. This means that her request has been balanced and it has finally arrived to an available front-end server (there are about 25 front-ends servers in the system).

Between the load balancers and the front-ends there are also two more hardware devices: a web accelerator and two application firewalls. The first of these devices tries to compress all traffic sent by the server to the user, so that the resulting data need less bandwidth and could be delivered faster to the user. The firewalls add security to the communication process, avoiding execution of non-allowed actions inside the intranet.

4. OPNET SIMULATION SOFTWARE

There are several discrete-event simulation programs specially designed for network simulation. One of these programs is the open-source OMNeT++ [Varga; 2001]. After some preliminary studies, though, we decided to use OPNET for two reasons: (i) it seemed to be the most widely tested, used and documented software in the network simulation area [Chang; 1999] [Aboelela; 2003] [Brown, Christianson; 2004] [Qadan, Guizani; 2005], and (ii) a free license for academic and research use was available from the software developer.

OPNET is a very complete software composed by several modules. Opnet IT Guru, the main module, provides a Virtual Network Environment (VNE) that can model the behavior of an entire network (including routers, switches, protocols, servers, and individual applications). It can be scaled from Local Area Networks (LANs) to Wide Area Networks (WANs) formed up by thousands of workstations.

VNEs can be created using a special purpose GUI interface (that is, selecting appropriate hardware elements -such as workstations, servers, routers, switches, hubs, etc.- and then connect them together according to the desired topology). On the other hand, for existing networks the program can automatically perform this modeling process. The software provides models for the most popular brands of network communication hardware. OPNET modeling and simulation flow diagram is shown in Figure 2.

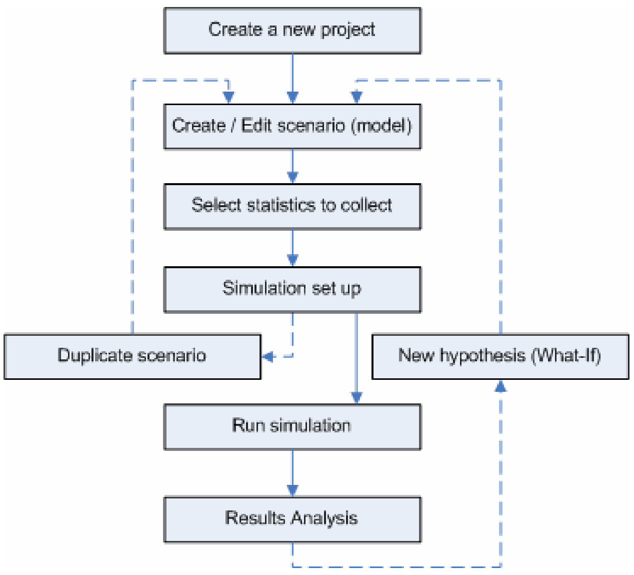

Figure 2: OPNET modeling & simulation flow

Another module, the Application Characterization Environment (ACE), allows modeling and analyzing the behavior of programs that reside above the topology layer –a web browser or a database, for instance. That is, the software can simulate both network topology and also any high level application that runs on that network.

For advanced research, OPNET Modeler offers advanced tools for model design, simulation, data mining and analysis. Using this software, it is possible to edit the source code of available hardware devices libraries. Modeler is based in a three-level design hierarchy: (1) a network model, where networks and sub-networks are defined; (2) a node model, where node's (hardware devices) internal structure is defined; and, (3) a processes model, where internal node states and functioning can be defined by using C/C++ programming [Svensson, Popescu; 2003].

5. SYSTEM MODELING

In this first stage of our project, we have focused in the partial modeling of the Castelldefels architecture. For that reason, we have centered our efforts into the Campus network, which is the infrastructure mainly used by students and professors.

Additionally, we reduced the model size of the Campus network, assuming that it had less servers and devices than the real system. In future experiments we expect to simulate the full model, that is: 25 front-ends, 3 mail servers, one backup load balancer and up to 25,000 concurrent users requesting HTTP, FTP and Email services.

The resulting OPNET model can be seen in **Figure 3**:

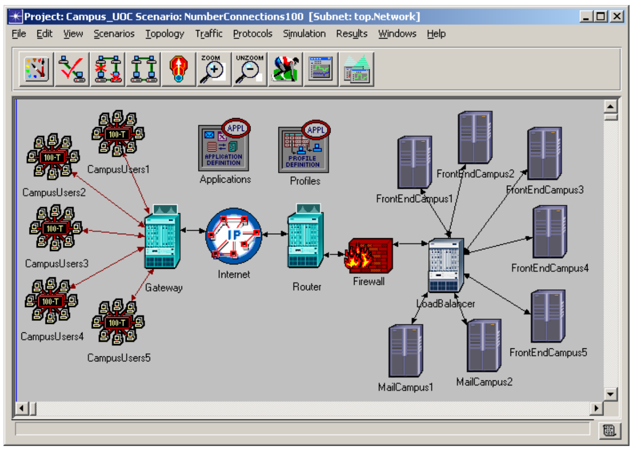

Figure 3: Castelldefels model implemented in OPNET

The main components of this model are the following:

- Applications: Each application defines a service that can be executed in a workstation or in a server. It also defines the load of the system. To emulate the behavior of the Campus application -a Web-based front-end with file transfer and email support-, we defined three OPNET applications: HTTP (Heavy Browsing), FTP (Low Load) and Email (Medium Load)
- Profiles: A profile is a group of applications to be used by some type of users such as students, managers, visitors, etc. Each profile also defines the statistical distributions for the simulation engine. In this first model, only one general profile -with support for all applications- was defined.
- Servers: There were 8 servers to execute applications: 5 front-end servers to emulate the basic Campus activity using HTTP and FTP, and 2 email servers. All servers were directly connected to the load balancer.
- Load balancer: It is a device that decides which server will attend the next user request. Servers are assigned to balanced applications. Each application is balanced using its own policy: Round-Robin for HTTP and FTP, and Number of Connections for Email.

All former devices were connected to a router through a firewall, which restricts the supported applications. The router was the single connection to the Internet.

- User Networks: several networks were setup to emulate students activity. Each network can have different number of users, from one to hundreds, and it was connected to Internet through a gateway. Services were directly requested to the Load balancer.

Some system devices were ignored in this preliminary model: several backbone and subnet switches, the database server and the redundant load balancer. In future models all of them will be integrated and fully supported.

6. SIMULATION RESULTS

Since this is just a preliminary study, we will show here only two of the multiple experiments we can make in order to validate the correctness of the model as well as its suitability for pedagogical purposes.

In the first experiment we wanted to:

a) Study the performance of a single-server system under different loads (number of connected users); the response time was expected to get worse as the number of users increases –since more users implies more requested services
b) Compare results in (a) with those of a multiple-server system under balancing policies.

Ten hours of activity where simulated. The simulation was done considering four scenarios. The first three scenarios had a single server with different number of active sessions (25, 50 and 100). The fourth scenario had 100 users connected to 5 front-ends balanced using a random policy.

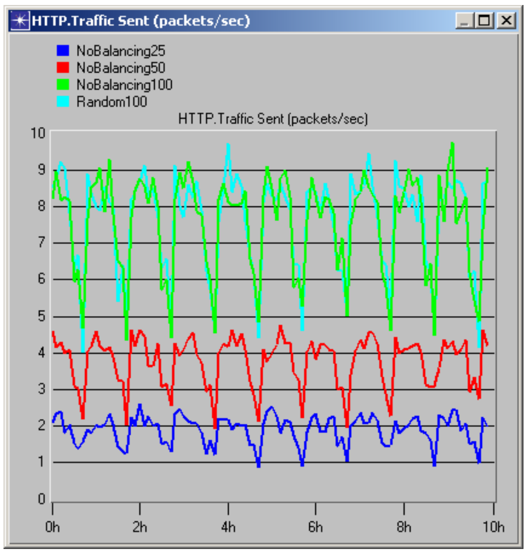

Figure 4: Experiment 1: Traffic sent (packets/sec)

As can be seen in **Figure 4**, as the number of users increases, more traffic is generated. Observe also that, as expected, introduction of load balancing has no effect on the level of generated traffic.

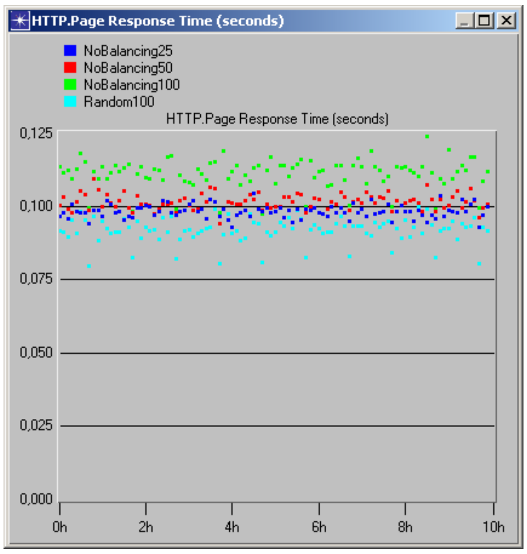

Figure 5: Experiment 1: Response time (in seconds)

As can be seen in **Figure 5**, use of load balancing for multiple servers can significantly improve performance. The response time associated to the highest load level under balancing is lower than any other single-server configuration (including the one with the smaller number of user sessions).

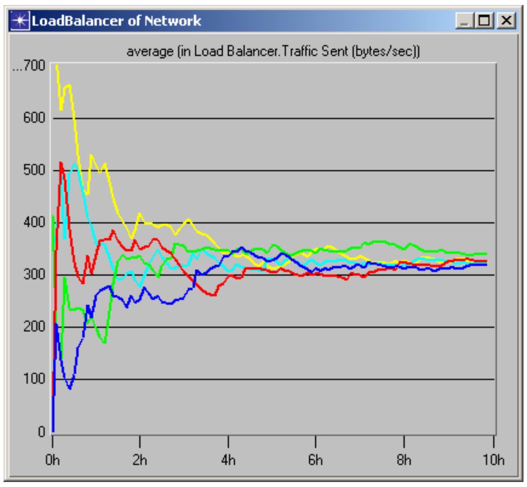

Figure 6: Experiment 1: Random balancing avg. traffic

Finally, **Figure 6** shows averages for traffic associated to each balanced server. After a warm-up period (more than 4 hours), all servers seem to converge to similar average loads.

The second experiment was designed to compare performance of different load balancing policies under the same user activity. Five servers were again balanced during ten simulated hours, with 100 users requesting HTTP services. The policies considered were: (a) Round-Robin (sequential), and (b) Number of Connections.

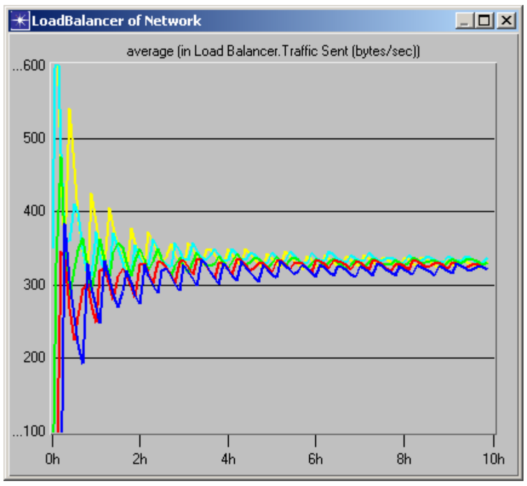

Figure 7: Experiment 2: Round-Robin average traffic

Figure 7 shows averages for traffic sent (in bytes/sec) by each balanced server under the Round-Robin policy. This policy assigns each incoming requests to the next server in the sequence. All servers tend to have the same average activity.

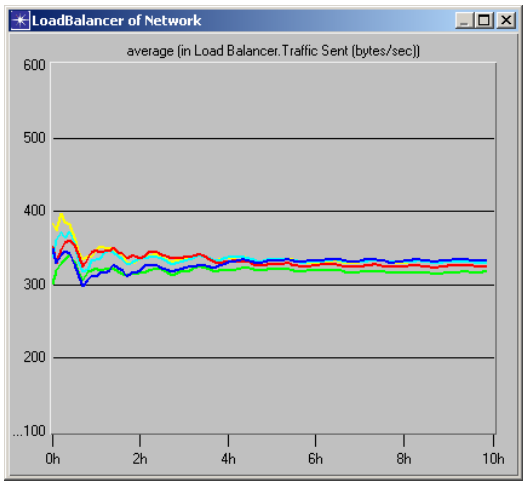

Figure 8: Experiment 2: Number of Conn. avg. traffic

Finally, **Figure 8** shows averages for traffic sent by all balanced servers under a Number of Connections policy. This policy seems to imply faster load stabilization for each server, but it also shows more variance among average servers load.

7. ACADEMIC APPLICATIONS

It is expensive to set up a physical networking lab for university students. Moreover, even if one university made that investment, such labs would have significant limitations when dealing with different possible scenarios (what-if analysis). In fact, it is virtually impossible to cover, using a physical network, the wide diversity of existing technologies and configurations. On the other hand, only LANs could be considered in a physical lab.

For that reason, use of discrete-event simulation, as a methodology to confront network design and fine-tuning problems, is not only interesting in the professional arena but also in the academic one **[Theunis, et al; 2003]**. Learning these technologies is really worthy: the current level of computer software and hardware allows the efficient application of simulation-based methods and algorithms to network analysis, allowing a major comprehension of networks' internal functioning process. Using simulation, students are able to analyze alternative scenarios and designs (what-if analysis) both for LANs and WANs.

8. FURTHER WORK

The model presented here is a preliminary one and, therefore, it needs to be improved so that it matches all subtle details of the real system (including inter-arrival time distributions, more detailed balancing policies session persistence policies, etc.).

CONCLUSIONS

We have discussed the importance of using probabilistic methods to study network performance. Among the available methods, simulation techniques offer clear advantages over analytical ones, such as: (a) the opportunity of creating models which faithfully reflect the real structure characteristics and behavior, and (b) the possibility of obtaining additional information about the system internal functioning. We have used OPNET to develop a preliminary model of the computer system that gives support to the UOC intranet. The model is still far from reflect all real system details, but it allow us to start experimenting with some what-if scenarios. More work is being developed to mimic the real system characteristics, so that derived results will help managers to take strategic decisions regarding the system. Finally, we have discussed how discrete-event simulation software can add value to the network engineers training process.

REFERENCES

- Aboelela, E. (2003): *Network Simulation Experiments Manual*. Morgan Kaufmann
- Banks, et al (2001): *Discrete-Event System Simulation*. Prentice-Hall
- Brown, K.; Christianson, L. (2004): *Opnet Lab Manual*. Prentice Hall.
- Chang, X. (1999): “Network simulations with Opnet”. In *Proceedings of the 1999 Winter Simulation Conference*, pp. 307-314
- Kurose, J.; Ross, K. (2005): *Computer Networking: A Top-Down Approach Featuring the Internet.* Addison-Wesley

- Law, A.; Kelton, D. (2000): *Simulation Modeling and Analysis*. McGraw-Hill
- Peterson, L.; Davie, B. (2003): *Computer Networks. A Systems Approach.* Morgan Kaufmann
- Qadan, O.; Guizani, M. (2005): *OPNET Lab Manual*. John Wiley & Sons
- Svensson, T.; Popescu, A. (2003): “Development of laboratory exercises based on OPNET Modeler”. Master thesis. Available at: http://jeep.its.bth.se/~adrian/opnet/
- Theunis, J.; et al (2003): "Advanced Networking Training for Master Students Through OPNET Projects". In *Proceedings of the 2003 OPNETWORK Conference*, Washington D.C., USA
- Varga, A. (2001): "The OMNeT++ Discrete Event Simulation System". In *Proceedings of the European Simulation Multiconference* (ESM'2001). June 6-9, Prague, Czech Republic.