Undergraduate Data Communications and Networking Projects Using OPNET and Wireshark Software

Vasil Y. Hnatyshin and Andrea F. Lobo
Computer Science Department
Rowan University
Glassboro, NJ 08028
1(856)-256-4805

hnatyshin@rowan.edu and lobo@rowan.edu

ABSTRACT

The national importance of creating "content reflecting cutting edge developments in STEM fields" has been recognized the National Science Foundation. The National Research Council has also noted that creating engaging laboratory and classroom experiences is a challenge to effective undergraduate education in STEM disciplines. In this paper we present our endeavors to address these issues in the field of computer networks. We discuss our experiences with novel laboratory and classroom projects using OPNET and Wireshark software products in undergraduate networking courses. These laboratories help undergraduate students understand fundamental networking concepts through modeling and simulation of computing systems. We have found that their use enhances student learning and interest in the subject of computer networks.

Categories and Subject Descriptors

K.323 [**Computers and Education**]: Computer and Information Science Education

General Terms

Design, Experimentation, Human Factors

Keywords

Data Communications, Networks, Computer Science Education Research, Laboratory Environments, Software Tools, Simulation, Modeling, OPNET, Wireshark

1. INTRODUCTION

"Computer science education is by its very nature an active learning discipline" [1]. This fact has been well documented in the literature [2 - 4]. The National Research Council has also noted that creating engaging laboratory and classroom experiences is a challenge to effective undergraduate education in Science, Technology, Engineering, and Mathematics (STEM) disciplines [5]. The national importance of creating "content reflecting cutting edge developments in STEM fields" [6] has been recognized. Our observation of today's undergraduate Computer Science (CS) students confirms this. We find that many of today's CS undergraduate students have spent many hours using a computer and the Internet, are task oriented, like to try things out, and are more interested in hands-on activities rather than a pure lecture class format.

The typical undergraduate networking course is taught as a lecture with homework and programming assignments. A primary student complaint against this approach has been the lack of hands-on experience [7]. Recently, more networking textbooks come with supplemental lab assignments to enhance the course [8-10]. In this paper we present our endeavors to develop and evaluate engaging laboratory and classroom projects to help undergraduate students understand fundamental networking concepts via the modeling and simulation of computing systems.

This paper discusses our experiences using novel laboratory projects using the OPNET [11] and Wireshark [12] software products in undergraduate networking courses. OPNET [11] is a leading software product for simulation and modeling of network applications, protocols, and technologies. It includes a vast library of communication devices, media, and protocols. OPNET Inc. provides free versions of some of its most popular products for educational use. Wireshark [12], formerly known as Ethereal, is a public-domain packet capture and protocol analysis software. It allows a user to capture live packets, apply filters, display packet headers in a convenient readable format, and generate basic collected trace statistics. Wireshark packet traces can be imported into OPNET for more advanced analysis or as a source of real traffic. The literature describes other experiences using networking simulation tools and packet analysis software in education [13-16]. However, there has been little or no research on the effectiveness of OPNET and Wireshark products in undergraduate networking courses.

The rest of the paper is organized as follows: Section 2 describes how OPNET and Wireshark software can contribute to undergraduate CS education. Section 3 describes these products in greater detail. Section 4 presents the contents of the projects. Section 5 discusses the effectiveness of the projects. The paper concludes with Section 6.

2. OPNET AND WIRESHARK AS UNDERGRADUATE LEARNING TOOLS

We have used OPNET and Wireshark in several offerings of the undergraduate networking courses at our institution since 2005. We have found that they help our students achieve some of the learning outcomes for our courses. In this section we present a list of these learning outcomes, followed by a discussion of the roles

SIGCSE'08, March 12–15, 2008, Portland, Oregon, USA.
Copyright 2008 ACM 978-1-59593-947-0/08/0003...$5.00.

played by OPNET and Wireshark in helping students achieve these outcomes.

The student learning outcomes for the undergraduate networking courses at our institution include:

1. Apply the scientific method by developing and executing a simulation, and analyzing its results;
2. Understand fundamental networking concepts;
3. Understand important Internet protocols; and
4. Analyze network system performance.

CC2001 states that the first learning outcome of applying the scientific method is a "vital component" of Computer Science [17; p. 46]. Conducting a simulation study is precisely applying the scientific method. A scientist conducts a simulation study to attempt to answer one or more questions about a particular system. The main steps in a simulation study are [18]:

1. Identify the objectives of the study,
2. Develop/design the simulation model of the system,
3. Set-up and configure the simulation,
4. Execute the simulation and collect data, and finally
5. Analyze and validate the collected results.

These steps map into the scientific method in the most fundamental sense: Steps 1 and 2 include hypothesis formation; Steps 3 and 4 describe conducting an experiment that can be replicated; and Step 5 is hypothesis confirmation or refutation. Networking experiments can be conducted using actual networks or using simulation software. The OPNET simulation software allows undergraduate CS students to simulate network systems with sophisticated applications and protocols. This alternative to a live networking laboratory is less time-consuming for the professor and more cost effective for the institution, can support a large number of students, and allows the study of a wide variety of networking systems.

The second and third targeted student learning outcomes are to understand fundamental networking concepts and important Internet protocols. We have used OPNET and Wireshark projects to study multiple networking systems with protocols that implement specific concepts during a one-semester course. For example, the students can verify the effectiveness of a three-way handshake or an Automatic Retransmission Request (ARQ) protocol, see how error conditions are handled, or confirm the details presented in the textbook or in a classroom presentation. The projects study important Internet protocols. OPNET and Wireshark also facilitate learning by discovery: For example, after learning about the implementation of *ping* using ICMP messages, the students can collect *traceroute* traces and encounter a similar idea; or they can run simulations to "discover" that some TCP flavors perform better than others when transmission error rates are high.

The fourth targeted student learning outcome is to analyze network system performance. Students can develop OPNET simulations to study system characteristics such as end-to-end delays for different applications, or throughput for different network conditions or protocol parameter settings. Simulation allows the students to easily study the system under many operating conditions and configurations, and develop their understanding of the performance impact of protocol parameter value selection.

Our students achieve other student learning outcomes as well, although these are not targeted objectives of our approach. Most obvious is the student ability to create and execute OPNET simulations and present collected statistics. While this is not an implicit objective of our undergraduate program, this skill may be very useful to some students in the workforce or graduate school.

From the teaching point of view, we have found that Wireshark and OPNET are useful tools. The students seem more attentive when we demo a live trace than when we discuss a static performance graph from the textbook, even when the ultimate result of the trace is essentially the same graph. We believe this is because a typical student can now understand how the data for the graph was obtained, and can recreate it if he/she needs further clarification. We can also require students to do a significant amount of relevant technical writing in the form of simulation study reports. We find that our students readily understand the need for this writing requirement. The simulation projects can also provide students with team experiences when several students work on a large assignment, and public speaking experiences when different students work on variations of a project and present their findings to the class.

3. THE OPNET AND WIRESHARK SOFTWARE PRODUCTS

OPNET IT Guru and Modeler are the most popular OPNET software products for educational use. OPNET IT Guru is a network simulation package which is relatively easy to use and learn. It allows a user to use point-and-click to create and configure simple or sophisticated network systems, and conduct simulations to study and analyze the system's performance. OPNET Modeler has additional functionality that allows the user to modify existing system components and create new ones.

A user working on a typical OPNET IT Guru simulation proceeds to complete the following steps:

1. Follow the Configuration Wizard instructions to create a new project and a first simulation scenario.
2. Point-and-click to configure the first simulation scenario:
   - Create the network topology,
   - Select and configure the relevant applications,
   - Create the user profiles to specify how the configured applications are used by the end systems,
   - Deploy applications by associating user profiles with the end systems,
   - Configure non-default parameters of the relevant protocols,
   - Specify the statistics to be collected during simulation,
   - Configure scenario parameters such as simulation duration, random number generator seed, etc.
3. Create a copy of the first scenario and modify the values of simulation parameters as needed. This step is performed for each scenario.
4. Execute the simulations for all scenarios.
5. Analyze the graphs and values of the collected simulation statistics.
6. Repeat step 2 through 5 until the results are valid.

Wireshark has a straightforward graphical interface through which a user can capture live packets, apply filters to select packets with desired characteristics, and display packet headers in a convenient readable format, and view trace statistics. A user working on a typical Wireshark task would:

1. Select a computer network interface from which to collect the trace.
2. Specify trace collection parameters such as duration, number of packets, filter, output file name, etc.
3. Start packet collection.
4. Analyze collected trace by examining packet header values, studying protocol hierarchy statistics or packet counts for certain protocols, following TCP streams, etc.

4. NETWORKING COURSES USING OPNET AND WIRESHARK

OPNET IT Guru and Wireshark are not trivial and may seem intimidating to some undergraduate CS students. Our experiences indicate that a classroom demonstration is valuable for most students. We found that it is also important to introduce the concepts of modeling, simulation, and packet header analysis. We dedicate one week of our semester to address these topics and demonstrate the use of both software products. The remainder of this section describes how we present these topics in class and the projects that we assign.

It is important for students to understand the role of modeling and simulation, and to be familiar with the steps for developing a simulation study, before introducing the OPNET software. We begin our discussion of OPNET by presenting the basic OPNET file and directory structures, the difference between a project and a scenario, what a network model is, what discrete event simulation is, and how to interpret and validate simulation results. Then we describe how to use OPNET, concentrating on: creating a project and scenarios, creating a network topology, creating and deploying network applications, configuring important network protocols and technologies, configuring and executing a discrete event simulation, and, finally, collecting and interpreting the simulation results. We present this material in 75-minutes ending with a demonstration of using OPNET to develop a simple simulation study from scratch. Since most OPNET simulations are not trivial, we found that our students benefit from an immediate homework assignment to create and run their own simulation in OPNET. This assignment does not ask students to analyze results but concentrates on various OPNET features. We start the next lecture by reviewing and providing another OPNET example.

Wireshark is much simpler software than OPNET and it can be introduced to the students in approximately 30 minutes. As with OPNET, we discuss the benefits of capturing and analyzing packets retrieved from the live network before describing the Wireshark software. Next we demonstrate how Wireshark operates and discuss its main components: the packet capture and packet analyzer modules. Then we introduce the Wireshark GUI: selecting a network interface from which the packets are to be collected, applying filters, collecting a live trace, viewing packet headers in the collected trace, and using various useful Wireshark features such as analyzing and viewing statistics of the collected trace. As with OPNET, we assign an immediate homework project in which students collect a live trace, apply filters, and perform simple protocol analysis (e.g. what was the number of IP, TCP, and UDP packets in the collected packet trace and why). An alternative approach suggested by Kurose and Ross [10], is to introduce Wireshark to the students via a homework assignment consisting of two parts: a reading section and practical section that requires students to perform certain tasks using Wireshark.

Several OPNET laboratory manuals are available [19 - 21], but we found that these assignments are difficult to adopt without major modifications. All OPNET laboratory manuals that we examined are written for OPNET versions 9.1 or 10.0 which are not compatible with version 12.0 that we use at our institution. Additionally, published OPNET laboratory manuals are written in the form of strict step-by-step instructions, specifying which button to press, what to type in, or which option to select. To complete such a laboratory assignment the student needs to meticulously follow the instructions but need not understand either simulation or the software. Furthermore, it is easy to make a mistake by skipping a step or having a typo, but then it is very difficult to identify this error.

Thus, we created our own set of OPNET laboratory projects that follows the topics of our course, teaches the students how to use OPNET, challenges the students to design and analyze, and illustrates and teaches various networking concepts. It was our goal to specify the projects in a manner that asks the student to perform a certain task (e.g. create the topology shown in a figure, set the TCP receiver window size, etc) without explicitly detailing which button to press or what to type. As the students gain more experience with the software, the instructions for subsequent projects become more independent of OPNET, and, eventually, only specify the objectives of the study and a high-level system configuration (such as network topology, protocol configurations, link capacities, etc). The students must analyze the objectives, design the simulation scenarios, configure the OPNET simulation, and carry out the study. The students also determine by themselves how to name projects and scenarios, which device models to select, which statistics to collect, etc. The latter, more abstract, project descriptions are particularly appropriate for OPNET since it allows multiple ways to configure a simulation system. Furthermore, OPNET Inc. releases new software updates frequently. The higher level project descriptions do not need to be updated when the software changes.

We are not aware of any Wireshark laboratory manuals published at this time because software recently changed its name from Ethereal. However, there are several Ethereal laboratory manuals available that can be adapted for Wireshark [22, 23]. For our courses, we created several new Wireshark laboratory assignments and adapted several Ethereal projects.

Figure 1 lists titles and brief descriptions of selected OPNET and Wireshark laboratory projects that were used in the *Data Communications and Networking* and *TCP/IP and Internet Technologies* courses. The first two OPNET projects provide either a complete or partial simulation model and only ask the students to update them. Starting from lab 4, students create a complete simulation model from scratch. In our experience, students easily grasp the concepts of creating simulation projects and scenarios, building network topologies, configuring protocols, selecting simulation statistics, and executing simulations. However, students often have a hard time learning how to configure applications and user profiles, how to deploy applications, and how to interpret collected results. We pay special attention to these topics by providing plenty of examples, reviewing various application and user profile configurations, and assigning homework projects with which the students practice these tasks. In practice, the students will perform these tasks in every OPNET laboratory project but we stress and provide careful feedback on these topics, especially in the first few OPNET projects.

1. **Intro to OPNET:** *Use an existing model to execute a simulation, collect simulation statistics, and analyze displayed results.*
2. **Intro to Wireshark:** *Collect live trace, apply filter, and examine TCP, IP, and UDP headers.*
3. **Application layer (OPNET):** *In a given simulation model, configure and deploy a specified application, and examine its performance within a given network topology.*
4. **HTTP (OPNET):** *Conduct specified simulation study. Deploy HTTP application and examine the performance of various configurations:*
   - *HTTP 1.1 vs. HTTP 1.0 (e.g. persistent vs. non-persistent)*
   - *HTTP w. pipelining vs. HTTP w/o pipelining*
   - *Internet Explorer vs. Netscape*
5. **HTTP (Wireshark):** *Study GET and PUT commands and requests with embedded objects and enabled security features.*
6. **Transport Layer (OPNET):** *Create simulation study. Examine and compare performance of TCP and UDP protocols in a given network.*
7. **TCP (OPNET):** *Conduct simulation study. Tasks may include:*
   - *Examine influence of TCP on application performance by changing the receiver window, maximum segment size (MSS), Nagle's algorithm on/off, etc. Compare TCP flavors.*
8. **TCP (Wireshark):** *Collect a live trace and examine TCP headers. Collect statistics about how frequently various TCP options are used.*
9. **Network Layer, Internet Protocol (OPNET):** *Create simulation study and examine of influence of IP fragmentation on application performance (e.g. amount of data sent, the length of the path, the header size).*
10. **IP Addressing (OPNET):** *Conduct simulation study that contains a network topology divided into several subnets. Compute IP address assignment for the given network and use OPNET to verify IP address allocation.*
11. **IP, ICMP, ARP (Wireshark):** *Collect live trace and examine IP, ICMP, and ARP packet headers. Determine implementation of ping and traceroute application by examine ICMP packets collected in the trace. Examine ARP operation by modifying and clearing ARP table and viewing ARP packets collected in the trace.*
12. **Routing (OPNET):** *Examine performance of routing protocols by selectively breaking and restoring links in the network.*

Figure 1. Selected OPNET and Wireshark lab assignments

5. EFFECTIVENESS OF THE PROJECTS

We have used OPNET and Wireshark projects in several offerings of undergraduate networking courses for Computer Science majors at our institution. We have used various subsets of these projects in our *Data Communications and Networking* and *TCP/IP and Internet Technologies* courses since the spring of 2005. The *Data Communications and Networking* course introduces networking concepts and the functionalities of all layers. The *TCP/IP and Internet Technologies* course focuses on the Internet and the TCP/IP protocol suite. In our first attempt to use such assignments we relied on published OPNET laboratory manuals [19, 20]. Informal feedback indicated that students liked the class and enjoyed working with OPNET and Ethereal. However, the students frequently complained about the quality of OPNET laboratory assignments. Their main concern appeared to be the monotony of following step-by-step OPNET configuration instructions, the difficulty of debugging the mistakes and getting valid results, and the complexity of analyzing the collected statistics. Student feedback helped us re-think the way we prepared the OPNET and Wireshark laboratory projects, and how these products are introduced in class. We extended the class time dedicated to the introduction and overview of the software tools and we completely re-wrote the lab manuals.

Table 1 summarizes the formal student feedback, from ETS' Student Instructional Report II surveys [24], for the courses in which we used OPNET and Wireshark. The first four rows in Table 1 contain average student scores on a 0-5 scale, and the last two rows contain the percentage of students who answered "very effective" or "effective" to the corresponding questions. Table 1 shows that, as we gained experience and enhanced the OPNET and Wireshark projects more students reported that their learning in the course improved, that the course actively involved them in what was learned, that they were challenged by the course, and that the assignments were helpful in understanding the course concepts and material. The responses to the question "*did the interest in the subject area increased*" did not change much over time, but they are all close to 90th percentile[1] among Computer and Information Sciences courses in four-year undergraduate institutions [25]. We believe that such high student evaluations are a direct result of incorporating the OPNET and Wireshark projects into the course.

Table 1. Summary of student feedback

	"Much more" or "more" than other classes		
	2005[2]	**2006**	**2007**
Learning increased in the course	3.82	3.91	4.14
Interest in subject area increased	4.18	4.09	4.14
The course actively involved me in what I was learning	4.00	4.27	4.43
I was challenged by the course	3.91	4.00	4.00
	"very effective" or "effective"		
Laboratory exercises for understanding course concepts	36%	91%	100%
Helpfulness of assignments in understanding course material	45%	91%	100%

The effectiveness of the projects based on OPNET and Wireshark with respect to student achievement of the target student outcomes was assessed by the participating faculty members. We reviewed the final project submissions for evidence of student achievement of the first outcome which is to apply the scientific method. We found that 100% of the submissions successfully conducted a sound simulation study. A part of the final project was to analyze the performance of an assigned system, the fourth target student outcome. We found that 100% of the submissions provided a satisfactory analysis of system performance and thus achieved the fourth outcome.

The second and third target student outcomes are to understand fundamental networking concepts and important Internet protocols. We used the relevant questions on each course's final exam to assess these outcomes. These questions were generally more difficult than those in previous offerings of the same course. The percentage of satisfactory responses to the questions relating to second and third outcomes were 78% and 77%, respectively.

The final project for the second and third course offerings of table 1 required that the students design and analyze the simulation

[1] 90th percentile is 4.18. 80th percentile is 4.00

[2] 2005 and 2007 correspond to *TCP/IP and Internet Technologies* course taught in spring 2005 and 2007, respectively. 2006 corresponds to *Data Communication and Networks* course taught in Fall 2006.

model based on a high-level description of a networking system. In all cases the students were able to complete the simulation set-up portion of the project; in fact, these large projects were treated as routine by the faculty and students, and the simulation set-up portion of the project was assigned 10% or less of the grade. This is evidence that the students achieve the additional outcome of learning OPNET.

6. CONCLUSIONS AND FUTURE WORK

In this paper we discussed a novel way to enhance undergraduate networking courses via the implementation of hands-on, engaging projects using OPNET and Wireshark software. We found that using the projects was effective in helping students achieve the target outcomes. The formal student course feedback obtained from Student Instructional Report II surveys shows that the projects are engaging and contributed to student learning. We have found that their use in the classroom and the laboratory enhanced student learning and interest in the subject of computer networks. We are currently working on further refinement of the project descriptions, and possible uses of other OPNET products in education.

7. REFERENCES

[1] McConnell J, Active and cooperative learning: tips and tricks (part I), *ACM SIGCSE Bulletin*, v. 37, n. 2, June 2005

[2] McConnell J, Active learning and its use in computer science, *ACM SIGCSE Bulletin*, v. 28, n. SI, p.52-54, 1996

[3] Grissom S, Van Gorp M, A practical approach to integrating active and collaborative learning into the introductory computer science curriculum, *Journal of Computing Sciences in Colleges*, v. 16, n. 1, p.95-100, Oct 2000

[4] Briggs T, Techniques for active learning in CS courses, *Journal of Computing Sciences in Colleges*, v.21 n.2, p.156-165, December 2005

[5] "Evaluating and Improving Undergraduate Teaching in Science, Technology, Engineering, and Mathematics," 2003, http://www.nap.edu/books/0309072778/html/, accessed 9/5/2007.

[6] National Science Foundation, *Course, Curriculum and Laboratory Improvement Program Solicitation*, 2007, http://www.nsf.gov/pubs/2007/nsf07543/nsf07543.htm, accessed 9/5/2007.

[7] Smith R, The computer networks course: will it all fit?, *Journal of Computing Sciences in Colleges*, v.17 n.2, p.244-252, December 2001

[8] Comer D, Computer networks and Internets (4th ed.), Prentice-Hall, Inc., Upper Saddle River, NJ, 2004

[9] Stallings W, Computer Networking with Internet Protocols and Technology, Prentice-Hall, Inc., Upper Saddle River, NJ, 2004

[10] Kurose J, Ross K, Computer Networking: A Top-Down Approach Featuring the Internet (3rd ed.), Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 2005.

[11] OPNET Technologies, Inc. http://www.opnet.com/, accessed 9/5/2007.

[12] Wireshark: A Network Protocol Analyzer, http://www.wireshark.org/, accessed 9/5/2007.

[13] Riabov V, Challenging projects and virtual labs in web-enhanced networking technology classes, *Journal of Computing Sciences in Colleges*, v. 21, n. 6, June 2006

[14] Dixon M, McGill T, Karlsson J, Using a network simulation package to teach the client-server model, *ACM SIGCSE Bulletin*, v.29 n.3, p.71-73, Sept. 1997

[15] Goldstein, C., Leisten, S., Stark, K., and Tickle, A. 2005. Using a network simulation tool to engage students in active learning enhances their understanding of complex data communications concepts. In *Proceedings of the 7th Australasian Conference on Computing Education – Volume 42* (Newcastle, New South Wales, Australia).

[16] Carniani E, Davoli R, 2001. The NetWire emulator: a tool for teaching and understanding networks. In *Proceedings of the 6th Annual Conference on innovation and Technology in Computer Science Education.* ITiCSE '01. ACM Press, New York, NY, 153-156

[17] ACM/IEEE Joint Task Force, *Computing Curricula 2001 Computer Science Volume*, 2001, http://www.sigcse.org/cc2001/, accessed 9/05/2007.

[18] Banks J, Carson J, Discrete-Event System Simulation, Prentice-Hall Inc., Englewood Cliffs, NJ, 1984

[19] Brown K, Christianson L, OPNET Lab Manual to Accompany Data and Computer Communications 7th ed. and Computer Networking with Internet Protocols and Technology 4th ed. by William Stallings, Pearson Prentice Hall, Upper Saddle River, NJ, 2005.

[20] Christianson L, Brown K, OPNET Lab Manual to Accompany Business Data Communications 5th ed. by William Stallings, Pearson Prentice Hall, Upper Saddle River, NJ, 2005.

[21] Panko R, OPNET Labs to Accompany Business Data Networks and Telecommunications (4th ed.) by R. Panko, http://www.opnet.com/university_program/teaching_with_opnet/textbooks_and_materials/itg_panko.html, accessed 9/5/07

[22] Matthews J, Computer Networking: Internet Protocols in Action, John Wiley & Sons, Inc., Hoboken, NJ, 2005

[23] Ethereal Labs: Supplement to Computer Networking: A Top-Down Approach Featuring the Internet, 3rd ed. J. Kurose and K. Ross, http://www-net.cs.umass.edu/ethereal-labs/, accessed 9/5/2007.

[24] ETS, http://www.ets.org, accessed 11/15/2007

[25] Student Instructional Report (SIR II). Comparative Data. Four-year colleges and Universities, http://www.ets.org/Media/Products/Sircomp4.pdf, accessed 11/15/200