Resources for Teaching Computer Networks

Linda B. Lankewicz
University of the South
Sewanee, TN 37383-1000 USA
llankewi@sewanee.edu

Abstract

Teaching a computer networks course can be a challenge both in terms of deciding on the focus of the course and providing appropriate practical experiences to complement the theoretical issues. However, because the course overlaps other areas of the computer science curriculum, it is an opportunity to apply some of the concepts from other areas and to expose students to the research in the field. This paper describes a computer networks course to accomplish this along with a collection of resources.

Introduction

A course in computer networks spans significant areas of the computer science curriculum including architecture, operating systems, distributed processing, and the ethical use of computing systems. It also treats many of the "recurring concepts" in the *ACM/IEEE Computing Curricula 1991* [1], such as conceptual and formal models, complexity of large problems, efficiency, levels of abstraction, and security. Material in networks textbooks covers ATM, telephone, and wireless systems as well as LAN's and internetworking. An instructor must decide where to place the emphasis. Should one widely used protocol model such as TCP/IP be covered in depth or should equal treatment be given to several different models? How much time should be spent on issues at each layer in the protocol hierarchy? Should students thoroughly understand how physical transmission occurs in copper wire, fiber optics, microwaves, and satellites? Or should more time be devoted to data compression, audio and video, clients and servers, and Java?

The answer depends upon the preparation of the students and the resources available. I favor an emphasis on TCP/IP because it is the predominant model. The course also is an opportunity to expose students to networking research and to improve their skills in presenting research findings. I assign some topics for outside research, allowing more of the class time to be spent on the details of the data transmission path, following a message down the sender's protocol layers, across the network, and up the receiver's protocol layers. And more time can be spent on the problems encountered in delivering the transmission: error detection and correction, routing algorithms, congestion control, reliability, and security.

A computer networks course should include both theoretical issues such as these and practical experiences. Networks textbooks are filled with acronyms, but courses solely devoted to terminology have shortlived utility. Practical experiences are necessary to fully understand problems such as congestion control and performance measurement. For example, it is easy to recognize the importance of providing security for network communications, but the difficulty of doing so is not readily apparent. To realize the vulnerability of a computing system to a sequence number attack, one must know how routers function, why sequence numbers are used, and how sequence numbers are generated. Thus, in addition to selecting appropriate material for a course from the wide array of topics, an instructor must design assignments that are suitable for the level of the students and meaningful in terms of contributing to understanding. Sources of material and sample assignments that have been successfully used in computer networks courses are described in the sections that follow.

Programming Projects and Exercises

Programming with Unix sockets[1] gives students experience with protocols and communication between processes without having to deal with machine dependencies at lower levels. Students encounter different challenges than in their programming courses. Students must consider race conditions, concurrent processes, mutual exclusion, and other details of interprocess communication. If students have no previous

SIGSCE 98 Atlanta GA USA

[1] If Unix is not available, LinuX is Unix-like operating system for x86, Motorola 68k, Digital Alpha, Sparc, Mips, and Motorola PowerPC machines. See *http://www.ssc.com/linux/what.html*. The Macintosh version, MkLinuX, is available at *http://www.mklinux.apple.com/*.

experience with socket programming, they can be given samples of code to be modified in the initial assignments.

The following is a series of assignments that develops skills in programming interprocess communications.[2] After experience with the first four assignments, students have a greater understanding of how communication services such as *FTP*, *telnet*, *rsh*, *SMTP*, and *NNTP* function since they are implemented with sockets. The fifth assignment is an programming project.

1. Given code for a server which receives a message using stream sockets, create a client process to send a message to that server.
2. Convert the code to use datagram sockets.
3. Convert stream socket code to send a message from a client to a server on a different machine.
4. Convert the datagram socket code to send a message from a client to a server on a different machine.
5. Typical programming projects:
   - implement access controls at the record level for a small database maintained by the server and accessed by multiple clients
   - implement a talk program for use by multiple users on a network and discuss the considerations for supporting more than two users at once
   - implement a simple version of *FTP* called *SFTP*, supporting the commands *ls*, *send*, and *get*.
   - implement a Kerberos-type algorithm for authentication of communication between two clients with keys generated by the server
   - create a server application that spawns (forks) concurrent processes to communicate with clients and compare its performance with one that handles tasks sequentially
   - implement datagram sockets with and without providing reliability (acknowledgements and sequence numbers) and compare the performance

In the first assignment, students are given code for the server implementing calls to *socket*, *bind*, *listen*, *accept*, *read*, and *close*. They must read the documentation (*man* pages) and understand how these calls function. Then they are required to write code for a client to communicate with this server. Doing so involves learning to use *connect* and *write* calls.

For the second assignment, students are again given the code for the server. They are asked to explain how server's datagram *socket* calls differ from those for stream sockets. They must understand how to use *unlink* and learn to use *sendto* in place of *connect* and *write* in order to implement the client. The third and fourth assignments involve communication between client and server on separate machines.

Item five above lists typical programming projects that have been used as the advanced programming assignment. Students are required to complete the assignment in two parts as shown in Table 1. Including specific details when making the assignment helps students learn how to organize a project. One of the most important aspects is requiring that students state assumptions and constraints when they turn in the first part, the prototype. This helps them define the project so that it is reasonable and manageable.

Table 1: **Interprocess Communication Assignment**

Features the project must have:
server and client processes
more than one concurrent client
servers and clients on different machines
buffers for messages
a means of identifying messages
some form of handshaking
Evaluation of the project:
complies with requirements
satisfies statement of purpose
functions correctly
Part 1 - Prototype:
statement of purpose
description of features:
message headers
buffers (variable/fixed size, local/global, circular/list)
list of assumptions and constraints
prototype code with limited features
Part 2 - Completed project:
code and documentation

Some textbooks [15, 12] provide simulation programs for comparing various protocols at the host-to-network or internet layers. They can be used to experiment with error rates, packet loss, and priorities. A typical assignment might ask for a comparison of the timeouts and payloads delivered for two protocols. Students would run the simulator, plot the results in a graph, and state the conclusions reached. A typical assignment using simulation materials is shown in Table 2.

Additional exercises may require that students examine network configurations, determine the route taken for messages using programs such as *traceroute*, and conduct experiments to measure performance. They can discover how the decentralized management of the large network functions. For example, the file *netinfo/root-servers.txt*, available at *ftp.rs.internic.net*, lists the sites at the root level for name servers. (Each local name server must know the name of at least one of these.)

[2]Further details and samples are on the author's webpage, *http://lankewicz.sewanee.edu/lblweb/cs411.html*.

Table 2: **Simulation Assignment**

Performance Exercises:
State a premise to be tested.
List the parameters.
(lost packet rate, payloads delivered, priorities, timeouts, protocols, errors)
Run the simulator to test the premise.
Plot the results in a graph.
State the conclusions reached.
Evaluation of the project:
Premise statement: clarity, importance, feasibility, testability
Test runs: completeness, explanations, varying conditions
Conclusions: validity

The file *nsfnet/statistics/history. netcount*, available by anonymous *ftp* from *nic.merit.edu*, contains the number of networks announced each year. Students can be asked to calculate the maximum number of network identifiers in each class, plot the number of network addresses assigned each year, and predict when all of them will be used. After examining these figures, the class might read RFC 1917,"An Appeal to the Internet Community to Return Unused IP Networks (Prefixes) to the IANA (Internet Assigned Numbers Authority)."

Research

Students need experience locating relevant materials, organizing it to convincingly present the most important topics, formulating an opinion based upon the research, and defending the position taken on a subject. As pointed out in [4],"Successful research requires many skills and techniques different from successful studying ...we need to provide explicit guidance in the key concepts and skills particular to research. ..." Research should be a part of upper-level courses in the computer science curriculum, and a computer networks course offers variety of topics to be explored as shown the sample reading list in Table 3.

Long before students are ready to submit material to journals, they need to write papers in the style required by journals. They also need to understand the peer review process and to recognize how material can be improved through this process. To accomplish peer review in a course, papers without the students' names can be distributed to classmates with a review form. Each student then receives feedback from the instructor and classmates without knowing the source of the reviews. Each paper must be rewritten and resubmitted with revisions based upon the recommendations of the reviewers. For the final submission, the reviewers' comments must be used to improve the paper. Rewriting makes students aware of the need to improve their communication skills.

Table 3: **Sample Reading List**

Title	Topics
Communication and Control. Networks and the New Economies of Communication [9]	rethinking public control, limits of free flow of information, transnational corporate networks
Global Networks: Computers in a Sustainable Society [16]	environmental problem monitoring, broadcasting findings
Improving HTTP Latency [11]	improving performance of the HyperText Transfer Protocol
NCSA's World Wide Web Server: Design and Performance [6]	Web access patterns, performance strategies
The New WAN [13]	price of connectivity, impact of ISDN, SMDS, and ATM
Rights and Responsibilities of Participants in Networked Communities [3]	legal considerations for networks, free speech, privacy, intellectual property
Securing the Information Infrastructure [7]	security
TCP/IP Illustrated [14]	programming internals
Telecommunications Network Design Algorithms [5]	design issues, performance analysis, modeling algorithms

In addition to the research paper, students may design webpages to combine instruction, demonstrations, and links to further information. Not only must the pages be well written, but they must be organized to support users with different levels of expertise. It is a challenge to develop material that communicates technical details in a manner that is informative and instructional. Online webpages subject students to peer review in a global sense since they receive feedback, both positive and negative, from Internet users. Examples of undergraduate student webpage projects may be viewed at *http:// llankewicz.sewanee.edu/ lblweb/ cs411.html*. The topics covered are listed below.

- Web Robots and Search Engines
- Sound File Formats
- Privacy
- Password Authentication
- Java

Students can bring their own interest areas into projects for the course. The pages on sound file formats, for example, were created by a music major minoring in computer science.

Ethical issues can be debated effectively in a computer networks course. Because these issues require an understanding of the technical aspects of the problems, a debate or discussion should be preceded by studying background topics. For example, before considering the question "Should employers restrict employees' access to materials on the Internet?", it is necessary to understand the feasibility of enforcing such a restriction. Topics for deliberation include equal acess to technology, electronic vandalism, privacy, free speech, government controls, and risks associated with computing systems. These and other topics are discussed in [2, 3, 8, 9, 10, 16].

Students are interested in trends and developments in the computing field. Giving them an opportunity to share topics with the class makes the course more interesting. The network RFC's (Requests for Comments)[3] are an excellent source of material. Students can learn of historical development of network features by reading the older RFC's and of more recent developments in the newer postings. Some RFC's are important because they define standards. They provide details to supplement textbook topics such as router and host requirements, IP numbers, and protocols. Examples are RPC's 1340, 1600, 1122, 1123, and 1009. These and other useful RFC's are listed in Table 4.

Table 4: **Sample RFC's**

RFC	Subject
1984	Cryptographic Technology
1980	HTML Image Maps
1938	One-Time Password System
1935	What is Internet Anyway?
1925	The Twelve Networking Truths
1922	Chinese Character Encoding
1917	Return Unused Network Prefixes
1898	CyberCash Protocol
1855	Netiquette Guidelines
1948	Sequence Number Attacks
1600	Internet Official Protocol Standards
1340	Assigned Numbers
1122, 1123	Host Requirements
1009	Router Requirements

Another resource is the newsletter of the InterNIC Support Services. It has monthly figures for performance on the Internet and articles on various topics. Current and past issues of InterNIC News are available at *http://rs.internic.net/ nic-support/nicnews/.*

[3]RFC's may be obtained by anonymous *ftp* to *ds.internic.net.* The index is contained in the file *rfc-index.txt.* RFC's also are available in HTML form at various Internet sites including *http://sunsite.auc.dk/RFC/rfc/.*

Conclusions

Much has been written about the need to change the teaching format to accommodate the learning styles of students. Doing so involves more than reducing lecture time or incorporating more visual materials in classroom presentations. It involves changing the level of participation of students, increasing expectations and requirements for student performance, and making students more responsible for their own learning.

To accomplish this, a course may include a variety of experiences: textbook exercises, research papers, readings and discussions, programming projects, interpreting simulation results, webpage creation, and debates of ethical issues. Students need to develop an approach to learning that includes reading the literature in the field, critically evaluating information, and assimilating materinal in a form that is meaningful. They need experiences which will enable them to learn to do these things well. All of these cannot be accomplished in a single course, but each course should develop some competence in these areas. If the student is not challenged in this manner, we have not contributed to educating the student.

References

[1] ACM/IEEE-CS. Computing curricula 1991. *Communications of the ACM 34*, 6 (June 1991).

[2] BHIMANI, A. Securing the commercial internet. *Communications of the ACM 39*, 6 (June 1996).

[3] DENNING, D. E., AND LIN, H. S., Eds. *Rights and Responsibilities of Participants in Networked Communities.* National Academy Press, 1994.

[4] FEKETE, A. Preparation for research: Instruction in interpreting and evaluating research. *SIGCSE* (February 1996).

[5] KERSHENBAUM, A. *Telecommunications Network Design Algorithms.* McGraw-Hill, 1993.

[6] KWAN, T. T., AND MCGRATH, R. E. NCSA's World Wide Web server: Design and performance. *IEEE Computer* (November 1995).

[7] LUNT, T. Securing the information infrastructure. *Commmunications of the ACM 39*, 6 (June 1996).

[8] CAUSE TASK FORCE. *Privacy and the Handling of Student Information in the Electronic Networked Environments of Colleges and Universities* . CAUSE, 1997.

[9] MULGAN, G. J. *Networks and the New Economies of Communication.* Guilford Press, 1991.

[10] NEUMANN, P. G. *Computer related risks.* Addison-Wesley, 1995. (Current *Inside Risks* column can be viewed at *http://www.csl.sri.com/insiderisks.html).*

[11] PADMANABHAN, V. K., AND MOBUL, J. C. Improving HTTP latency. *Computer Networks and ISDN Systems* (December 1995).

[12] PETERSON, L. L., AND DAVIS, B. S. *Computer Networks, A Systems Approach* . Morgan Kaufmann, 1996.

[13] SALAMONE, S. The new WAN. *Byte* (May 1996).

[14] STEVENS, W. R. *TCP/IP Illustrated*, vol. 1, 2, 3. Addison-Wesley, 1994.

[15] TANENBAUM, A. S. *Computer Networks*, third ed. Prentice Hall, 1996.

[16] YOUNG, J. E. *Global Networks: Computers in a Sustainable Society*. Worldwatch Institute, 1993.