Teaching Experiences in Unix System Programming

Chia-Tien Dan Lo, Witawas Srisa-an and J. Morris Chang
Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616-3793, USA, (dan, witty, chang@csl.iit.edu)

Abstract: In this paper, we share the experiences of teaching Unix system programming at Illinois Institute of Technology (IIT). In IIT, the prerequisites for this course are *System programming* and *Operating systems* where system programming course focuses on the Microsoft Windows operating system. Consequently, most of the students are greenhorns to the Unix operating system. On the other hand, it contains almost every mandatory element in the realm of operating systems. For a single-semester course, indeed, it is hard for the students to master all the material and for the instructor to convey all the ideas clearly. To conquer these hurdles, firstly, we unclog the communication channel among students, the T.A. and the instructor by creating a proprietary discussion forum. Secondly, to save the time for taking notes and concentrate on the lecture, all lecturing slides, notes and examples are posted on the class Web site. Lastly, a project demonstration and presentation is required for every student. The project topics are carefully designed in order to cover most of the important concepts. Students can learn more by doing different project topics and it also avoids plagiarizing. Based on our experiences, all students can complete all the projects and get a whole picture of Unix system programming after taking the course.

Keywords: Unix, Operating System, System Programming, Teaching Experiences

1. Introduction

It is undoubtedly uneasy to teach programming course in one semester. First of all, there are too many programming languages available such as Fortran, Pascal, C, C++, Ada, Prolog, Basic, Clipper, Foxpro, Delphi, Dbase, Perl, Java, etc. As a matter of fact, there are over 2000 computer languages [2]. Although most of them may share the same structure and functionality, one language may be succinct to one particular application but tedious to another. Sometimes, it may be even worse if the wrong language is selected for a project. For example, it would be easier to write a database program by using Foxpro rather than using C language, even though C language can be used to program any low level requirements and *de facto* most of compilers of computer languages are written in the C language. Among these programming languages, Pascal, C or C++ will normally be selected as a beginning programming course for a undergraduate curriculum because they are considered as structural programming (Pascal and C) and object-oriented (C++) programming. Secondary, different environments of a programming language frustrate some students when realizing the material taught in classroom. Getting hands-on experiences is very important after students finish the programming course. It is the ultimate goal for learning programming which is writing programs and running them correctly. How to match teaching material with programming environments is a challenge to the instructor as well.

Unix system programming, on the other hand, is a course that combines programming languages and Unix operating systems. An operating system is the interface between a bare machine and a user [1]. It manages system resources and provides commands for a user to operate a computer. Resources include memory, disk spaces, printers, central processing unit (CPU), etc. Commands or Shell commands are composed of executing a program, terminating a process, querying system information, compiling a piece of user program, debugging program, reading mail, running a browser, etc. The aforementioned items are all falling into the realm of Unix system programming. Unfortunately, Unix operating systems are varied from vender to vender though some Unix standards have been built such as ANSI C, POSIX and X/Open.

The paper is organized as follows. Section 2 describes teaching material and platforms. Section 3 illustrates student interaction. Section 4 depicts projects and examinations and section 5 concludes the paper.

2. Teaching Material and platforms

The first step to create a successful lecture is to find out the programming platform. The programming platform includes the machine specification, the version of the operating system. In the department of computer science at

IIT, there are eleven SGI Indy workstations, two SGI Indigo2 workstation, one SGI Power Challenge server. Table 1 shows the specification of these machines.

Table 1. Computing facilities at computer science department, IIT

Type of Machines	CPU (Speed)	Main Memory	L2 Cache	Disk Space	O.S. Version
SGI Power Challenge	4 x MIPS R4400 (150 MHz)	128 MB	1MB	20 GB	IRIX v5.3
SGI Indy	MIPS R4600 (132 MHz)	32 MB	512KB	512MB	IRIX v5.3
SGI Indigo2	MIPS R4400 (250 MHz)	64MB	2MB	2GB	IRIX v5.3

IRIX operating system has been developed since 1982 by Silicon Graphics. It is known for its high performance computing, advanced graphics and visual computing. IRIX is compliant with UNIX System V Release 4 and POSIX. However, the major problem in IRIX 5.3 is its lack of some implementations such as multithreaded programming support, remote procedure calls, etc. The newest version of IRIX is 6.5.6 in which bugs have been fixed and more features have been added such as the Pthread library etc. Table 2 shows the types of interprocess communication features in IRIX v5.3 [3]. For example, there are three different implementations of semaphores in IRIX operating system: POSIX, IRIX and SVR4. Among these implementations, IRIX semaphores are proprietary to IRIX O.S. and SVR4 semaphores are compliant to AT&T System V release 4 O.S. However, POSIX semaphores are not implemented in IRIX v5.3.

In addition to the above computing facilities, some PC based Unix implementation such Linux is a promising Unix compliant operating system. Because it is free, students can install it into their own machines and get root privileges to access and maintain their systems. Linux has been designed to be POSIX-compliant and supports most of the Unix standards. For educational purposes, it is mandatory to have it installed for a student who is learning Unix system programming, system administration, modifying system kernel, etc. Another choice is VMware [3] which is a virtual machine running on top of an operating system. In other words, for example, it allows Linux run on top of Windows NT or vice versa. By using Vmware or other equivalent tools, it can greatly reduce the workload in system administration.

Table 2. Types of interprocess communication in IRIX operating systems

Type of IPC	Purpose	Compatibility	Support in IRIX v5.3
Signals	A means of receiving notice of a software or hardware event, asynchronously	POSIX, SVR4, BSD	Yes
Shared Memory	A way to create a segment of memory that is mapped into the address space of two or more processes, each of which can access and alter the memory contents.	POSIX, IRIX, SVR4	No POSIX
Semaphores	Software objects used to coordinate access to countable resources.	POSIX, IRIX, SVR4	No POSIX
Locks, Mutexes, and Condition Variables	Software objects used to ensure exclusive use of signal resources or code sequences.	POSIX, IRIX	Only support memory and file locks
Barriers	Software objects used to ensure that all processes in a group are ready before any of them proceed.	IRIX	Yes
Message Queues	Software objects used to exchange an ordered sequence of messages.	POSIX, SVR4	No POSIX
File Locks	A means of gaining exclusive use of all or part of a file.	SVR4, BSD	Yes
Sockets	Virtual data connections between processes that may be in different systems.	BSD	Yes

The course material needs to be based on the supported features of the programming platform. It is not feasible to cover in detail all versions of Unix platforms in one semester. Instead, by selecting topics that can be fully practiced after classes, it will not only reduce some irrelevant material but also let the students focus on specific topics and finish their projects in one semester. Moreover, the key feature of Unix programming would not be missed because Unix operating system has been developed to be portable and standardized. The rationale behind this is that the same concept and the same methodologies can be easily applied to other Unix implementations if the student can master one Unix version.

The selected topics [6] used in IIT are Unix and Ansi standards, C++ Language Review, C++ I/O Stream Classes, Standard C Library Functions, Unix File Systems and Unix File APIs, Unix Processes, Unix Signals, Alarms and Timers, Remote Procedure Calls, Interprocess Communication, Semaphores, Messages, Shared memory, C Shell Programming, CGI and HTML programming and maintaining a large system using the Make utility. Even though a wide range of course materials has been adopted, the depth of each topic will be emphasized. Furthermore, it is worth noting that the teaching goal is to integrate every technology related to operating systems into a whole software system design.

3. Student Interaction

3.1 Teaching equipment

Due to the advanced technologies of broadcasting and Internet, since 1976, IIT has offered distance learning programs and courses via microwave through the William F. Finkl Interactive Instructional Television Network, known as IITV. Currently, there are over 1800 IITV enrollments each year in upper-division under-graduate and graduate courses in computer science, biological, physical, and chemical sciences, chemical engineering, electrical and computer engineering, environmental engineering, materials, mechanical and aerospace engineering and others. Moreover, in 1996, IIT began offering courses via videoconferencing between campuses and in association with partners in other regional areas in the State of Illinois. IIT now offers programs and courses via the Internet and through other PC-based technologies. A typical studio classroom at IIT is equipped with a set of cameras, a set of intercom, a set of multifunctional overhead projector and a couple of Television sets. The multifunctional overhead projector will collect signals from TV cameras, slides and VGA output of a computer with Internet connected. The intercom is used as a communication channel between an instructor and remote site students. Basically, the remote site students receive exactly the same treatment as those inside the studio. The only difference is that an instructor is hardly aware of the real-time response from remote site students. However, it is always stimulated to raise a question and wait for the response from remote site or inside the studio.

One of the major advantages with networking connected classroom is the ability to show interactive responses from a real machine through the network. Because the course has not been designed to have a laboratory session, this on-line demonstration of system operations is especially beneficial while guiding students where to start. For example, some students may not have any experiences logging into a Unix system, coding a piece of program, compiling it, executing it or debugging it. By using a notebook computer connected to the Internet and display whatever is shown on the monitor, the introductory material can be done easily in one teaching hour. Additionally, some hands-on experiences such as tips and caveats can be conveyed to the students. The form of the teaching technique is essentially another variation of conducting a laboratory.

3.2 Class Notes

A roughly dichotomy for the selected materials can be referential information and design principles. Prototypes for library functions and system calls fall into referential information. It is hard to memorize all the referential information but knowing how to get related information is much more important. An evidence for this is that a library call may be invalid for another system. So knowing the design principles and fundamental concepts must be the top priority in the learning process. Anyhow, taking notes in Unix system programming course is really time-consuming. It is extremely inconvenient and inefficient not only for the students but also for the instructor if all program must be written in class. The purpose is to show the idea and where to find the related information if a student wants to build a similar project. Consequently, all the class notes are posted on the class Web site. Students can download and print the class notes themselves. This will greatly improve both teaching and learning efficiency. Unfortunately, the method can only be applied to course materials such as referential information. For design principles, assistance from a blackboard or whiteboard is still needed. For example, it is hard to explain system calls

such as fork(), exec() and mmap() by using some snippets of program. To convey the ideas of the aforementioned system calls, it is much easier to do that with drawing some memory block diagrams on the blackboard.

3.3 Internet Discussion Forum

The interactions pertaining to the class can be categorized into student-student, student-T.A., T.A.-T.A., student-instructor and T.A.-instructor. It is inefficient for a T.A. or the instructor to answer a question for multiple times. On the other hand, a student's question may be others' as well. So the basic idea is to set up an Internet discussion forum is to minimize the cost of raising a question and answering a question. In addition to the above advantages, we have found that an Internet discussion forum has the following merits:

1. Students can learn from each other by posting a question and solving the question themselves. Especially a remote site student can share ideas with a main campus student with it. Moreover, a good student can have a way to help a moderate student.
2. A student can raise a question to the T.A.'s with prompt responses. Traditionally, the T.A.'s can be reached only in the office hours or through e-mail. These communication channels are more or less slow and inconvenient. The prompt response is particularly necessary when a student tries to figure out some bugs in his project.
3. The Internet discussion forum can also be used to coordinate the T.A. work in a course that has more than two T.A.s. Sometimes, it is hard to divide the T.A. work evenly if there are more than two T.A.s. The Internet discussion board can be functioned as a question queue. The question queue has the priority precedence: student, T.A., and instructor. That is a question will be answered by one of the T.A.s if no student can solve it. The instructor only needs to answer a question or correct a posted solution if none of the T.A.'s can handle it.
4. An instructor knows students' comprehension by reading the posted questions. T he information is useful for the instructor to adapt the future class material and polish teaching techniques. Also, some important questions can be considered for an extra lecture to clarify the idea. It may be called student oriented teaching.

Although setting up a discussion forum on Internet has the above advantages, there are some issues that need to be emphasized. The student may not show up in class but keep asking questions on the forum. To some extent, this is tolerable because the student wishes to learn. However, the situation can be worse because it consumes some resources. Moreover, some student may post a solution for an assignment before its deadline. Hopefully, to take the maximal benefit from the discussion forum, rules must be made at the beginning of a semester. For example, only ideas can be posted on the forum, questions related lectures and projects, etc. Based on our experiences, no big problems have occurred so far.

4. Projects and Examinations

For any programming course, giving an in-class examination is not applicable. The major reason is the procedure to write a piece of program: flow-chart design, coding, compiling, debugging, etc. Each of them needs interaction with a computer. Without a dedicated computer laboratory, the Unix system programming class does not ask for any examinations. Instead, three projects are required. The section shows our methodologies in conducting projects in this course.

4.1 Topic Design

The concerns in choosing topics are composed of theoretical perspective and practical training. Some fundamental problems can be related to classical problems such as Dinning philosopher problem, Reader Writer Problems, etc. The type of project topic can be a good practice in concurrent programming and inter-process communication mechanism. However, since they are well-known problems, they could have been practiced in some prerequisite curriculum related operating systems. It's better to modify the project specification of this type in a way that makes it more specific and more constrained. Otherwise, we may turn out to re-invent the wheel.

4.2 Coverage

One way to avoid plagiarizing is to have a different topic for every student. More project topics can cover more concepts and student can learn more by doing different projects. Nevertheless, to give a different topic is easy but to give an even workload may be difficult. We compromise the workload and different topics by assigning two same-topic projects and one different-topic project. Selected project topics in the course are listed in Table 3.

4.3 Larger Project

Larger projects need more effort and time and require several students to collaborate with each other. When dealing with a large project, students must divide workload into several equal pieces themselves, prototyping a unified interface, debugging subsystems, etc. This will force students to study themselves and learn how to cooperate with others. Good candidates for larger project topics are writing a Web server, FTP server, multithreaded Database server, etc.

Table 3. Selected project topics

1	Process creation	8	Guessing file types using magic number
2	Variable length argument function	9	Recursive execution of commands
3	Parallel programming – prime number by sieving	10	Chat program using RPC and TTY devices
4	Shell over Web	11	Mail delivery
5	Basic Database operations using C-shell script	12	Finger utility
6	Database server using named pipes and file locks	13	Bounded buffer problem
7	Two-directional pipe and manual pages	14	Dinning philosopher problem

To simulate industry practice, the concept of teamwork is fully emphasized. There may be a situation that only part of the team actually shares most of the workload. To overcome this drawback, the team leader must call a project meeting at least once a week to ensure that all members are contributing their individual work properly. Moreover, to encourage student in doing a decent project, a final project demonstration and competition can be held at the end of a semester. Consequently, a teamwork larger project may be a good substitute for small assignments and examinations.

5. Conclusions

Throughout this paper, we have shown our experiences in teaching Unix system programming at Illinois Institute of Technology (IIT). In IIT, the prerequisites for this course are *System programming* and *Operating systems* where system programming course focuses on the Microsoft Windows operating system. Consequently, most of the students are not familiar with Unix operating systems. Therefore, it is important to select suitable material for the beginners. At the same time, the material needs to include some advanced topics for more senior students. For more sophisticated students, we also assign an independent project. On the other hand, this curriculum contains almost every mandatory element in the realm of operating systems. For a single-semester course, indeed, it is hard for the students to master all the material and for the instructor to convey all the ideas clearly. To conquer these hurdles, firstly, we unclog the communication channel among students, the T.A. and the instructor by creating a proprietary discussion forum. Secondly, to save the time for taking notes and concentrate on the lecture, all lecturing slides, notes and examples are posted on the class Web site. Lastly, a project demonstration and presentation is required for every student. The project topics are carefully designed in order to cover most of the important concepts. Students can learn more by doing different project topics and it also avoids plagiarizing. Based on our experiences, all students can complete all the projects and get a whole picture of Unix system programming after taking the course. Complicated subjects such as interprocess communication, locking system and multithreaded programming can be absorbed within one semester.

6. References

[1] J. S. Gray, "Interprocess Communications in Unix, The Nooks & crannies," Prentice Hall PTR, Upper Saddle River, NJ 07458, Prentice-Hall Inc., 1998.

[2] Terrence Chan, "Unix System Programming Using C++," Prentice Hall PTR, Upper Saddle River, NJ 07458, Prentice-Hall Inc., 1997.

[3] D. Cortesi, "Topics in IRIX Programming," Document Number 007-2478-007, Sillicon Graphics Inc., 1996

[4] The Language List, http://cui.unige.ch/langlist

[5] Vmware, http://www.vmware.com/

[6] Chia-Tien Dan Lo, CS451 Class Web Site, http://csl.iit.edu/~cs451

9. Acknowledgements

With his valuable assistance, the authors would like to thank Therapon Skotiniotis, the teacher assistant of the class CS451 at Illinois Institute of Technology.