Journal of Engineering Education Transformations ,
Volume 31 , No. 3, January 2018, ISSN 2349-2473, eISSN 2394-1707

Teaching Operating Systems - Programming assignments approach

Shantala Giraddi[1], Priyadarshini Kalwad[2], Suvarna Kanakareddi[3]
[1,2,3]Department of Computer science and engineering,
BVB College of Engineering and Technology, Karnataka.
[1]shantala@bvb.edu
[2]priyadarshini@bvb.edu
[3]suvarna_gk@bvb.edu

Abstract: Operating systems classes always include a heavy lecture component to explain topics as operating systems architectures, synchronization, process management, memory management and file system. However in addition to these lectures students need some form of personal exploration to investigate how the concepts and algorithms are implemented. The authors report a series of programming assignments designed for operating systems course to impart the kernel programming experience to the students. Simulation of functionalities of operating systems, survey of contemporary operating systems, implementation of shell, that acts as OS interface to the user and addition of new API to an existing operating system. These projects are of moderate complexity but require the students to understand advanced concepts of operating systems. Results show that these assignment increase understanding level of core concepts and also expose students to complexity of a real operating system. The programming assignments were specially designed so as to make students explore the various components of OS. The authors found that these assignments definitely enhance the learning experience and there was a remarkable change in the learning level of the students as evident in the grades obtained by the students.

Shantala Giraddi
Department of Computer science and engineering,
BVB College of Engineering and Technology, Karnataka.
shantala@bvb.edu

Keywords: loadable kernel modules, system call, shell, kernel programming

I. Introduction

The authors made a survey of OS programming assignments at the top CS programs in the leading universities in the country. It was found that majority of course projects are simulation assignments.

A. Designing projects for operating system course:

Real operating systems are large software products. It is difficult to design projects involving these operating systems since it is difficult to understand them. Operating systems programming assignments can be either user level or kernel level. User level assignments can be simulations of various OS algorithms such as CPU scheduling algorithms, main memory management algorithms, disk scheduling algorithms. These algorithms run in user mode of operation. Although these projects give some insight into OS functionality but they do not provide the experience of kernel level development. Kernel level projects require writing code that runs in supervisor mode. To do kernel programming students must be aware of various modules of kernel, intermodule dependency, and various system variable/data structures to make even a small change in the kernel. In addition to this students should also know to download, compile, kernel testing debugging and rebooting etc.

B. Choice of operating systems:

Real production OS are quite complex. Operating systems like linux which are open source can be used for teaching purpose. Over the years they have fully grown, have many features available. Hence it is very difficult to design projects for the course with these systems. Many pedagogical small OS such as Nachos, XV6, XINU, OS161 and ECOS are widely available which can be used for such projects. Sample operating systems are listed in Table 1. Linux has the advantage of immense popularity and open source software base at the same time can give the student real world project experience. It has a lot of documentations available.

2. Linux Support For Kernel Programming - Lkm

The basic way is to add code to Linux is to add some source files to the kernel source tree. The kernel is recompiled, but this process is very time consuming. Another way is to add code to the Linux kernel while it is running. A module that can be added in this manner is called a loadable kernel module. They can be of one of these categories.

- Device drivers
- File system drivers
- System calls

Base kernel is the part of the kernel that is bound into the image that you boot, i.e. all of the kernel except the LKMs is called base kernel. LKM are very much part of the kernel. And LKMs communicate with the base kernel. LKM is called a "kernel extension."

We can add the module into the kernel by loading it as an LKM or binding it into the base kernel. LKMs have a many advantages over binding into the base kernel. One advantage is that we need not rebuild kernel after adding the module. This saves lot of time time and also possibility of introducing an error in rebuilding and reinstalling the base kernel is also reduced. One version of Linux, Tiny core 8.0 is only 16MB in size, released on April 10, 2017. It is better to leave base kernel it untouched once it is in working condition.LKM are modules that can be loaded and unloaded from kernel on demand. They provide an easy way to extend the base kernel as per new requirements. Device drivers are implemented in this way. All the device drivers are not required always. When those drivers are not needed, we can unload those specific drivers, which in turn reduces the kernel image size. The kernel modules will have a .ko extension. Table 2 lists the Linux utilities provided for kernel programming. The LKM advantages are,

Table 1. Popular pedagogical operating systems

1.	XINU	1. IIT Kanpur 2. Columbia university 3. Rutgers university 4. United states naval academy's 5. Universityof North Carolina chapel hill 6. Florida state university
2.	NACHOS	1. University of California, Berkeley 2. Loyola University Chicago 3. University of Chicago 4. UC Santa Cruz CMP 111 5. Oregon GraduateInstitute CSE 513 6. Virginia Tech CS3204 7. University Of Kansas 8. University ofGeorgia
3.	XV6	1. University of Illinois at Chicago 2. Rutgers University 3. Northeastern University 4. Yale University 5. Columbia University 6. Ben-Gurion University 7. Johns Hopkins University 8. Tsinghai University 9. University of Wisconsin-Madison, 10. Binghamton University 11. University of Utah 12. IIT Madras in India

a. LKMs are loaded when they are needed and therefore can save memory. All parts of the base kernel stay loaded all the time.

b. LKMs are much faster to debug and maintain.

LKMs run at the same speed as that of base kernel modules.

3. Xinu

XINU is an operating system for embedded

Table 2. Linux Utilities to Manipulate Kernel Modules

a.	lsmod : List Modules that are Loaded Already
b.	insmod : Insert Module into Kernel
c.	modinfo : Display Module Info
d.	rmmod : Remove Module from Kernel
e.	Modprobe: Add or Remove modules from the kernel

systems. It is developed by Douglas Comer for educational use at Purdue University. XINU stands for XINU Is Not UNIX . It has been ported to many hardware platforms like Sun-2 ,Sun-3 workstations, DEC PDP-11 , VAX, Intel x86, PowerPC G3 and MIPS. Since then, Embedded XINU has been ported to other platforms such as the QEMU, MIPSel virtual environment and the Raspberry. Laboratory environment and a lot of curriculum materials are already in use for courses in Operating Systems, Hardware Systems, Embedded Systems, Networking, and Compilers at Marquette University and other colleges/universities. There are two approaches to run XINU. A version of XINU that runs in a Virtual Machine (VM) environment is available, for users who prefer not to touch real hardware. This version is better, hardware remains safe and it can run on a conventional computer with no extra hardware. XINU provides very good platform for pedagogical activities and students were encouraged to explore XINU for assignments. Introductory session was conducted on XINU OS.

Students were given choice of using Linux or XINU for kernel programming activity. Linux being more popular among students, around 60% students chose Linux for kernel level project. 40% of students chose XINU.

SYSTEM CALLS. : System calls provide a mechanism to make use of services from the kernel. For example, there are system calls to open a file, to read a file, to create a new process. Most standard system calls are always built into the base kernel (no LKM option). But when we need a new functionality, we can invent a system call of our own and install it as an LKM. We can override/tweak an existing system call with an LKM of your own.

Possible new system calls can be taken from [6].

4. Related Work

1. Xiaohong Yuan et.al [1] used Visualization technology to graphically illustrate various concepts. Authors compared the usage of CAMERA and MLFQ visualization tools in different semesters, also compared student performance in those classes. MLFQ [3] is a software package that simulates the multilevel feedback queue scheduling algorithm for a single CPU. CAMERA (Cache and Memory Resource Allocation) [4], is a collection of workbenches for virtual memory. It provides users with simulations and interactive tutorials to help them better understand memory management concepts. 90% of the students considered both tools useful.

2. Jason Nieh and Chris Vaill [2] created a virtual kernel development environment. This environment provided a way in which operating systems can be developed, debugged, and rebooted in a shared computer facility without affecting other users. This platform is used along with a series of five Linux kernel level projects designed to enable the students to modify and replace subsystems of operating system. The projects were to build a kernel and install and boot it, second project is implementing a new kernel CPU scheduler, implementing a new access control list mechanism for the commonly used Linux ext2 or ext3 file systems and replacing the Linux kernel's page replacement algorithm,

3. Oren Laadan et.al [3] also used Linux based projects in Columbia University. Series of projects start include writing system calls to access process data, synchronization primitive mutex, modifying the existing Linux kernel scheduler, implement a framework to track the working set of processes.

4. Francis Giraldeau [4] discussed a set of novel execution visualization tools and experiments. Assignments were quite simple such as illustrating usage of fork and exec, Synchronization tools, signals and inter-process communication. All these assignments were based on more on visualization tools rather than programming.

The subsequent subsection elaborates on the programming activities, which are used by the authors in conducting the course.

5. Course Design

Operating system is a core 4-credit course in the 6thsemester.

Objective of the course were,

1. Understand the role that the operating system plays in the management of the various resources of computer system.

2. Discuss the basic issues in process management.

3. Distinguish between the different deadlock prevention and avoidance schemes.

4. Evaluative and analyse various algorithms in memory management

5. Modify existing open source kernels in terms of functionality

The first four course outcomes were concerned with making the theoretical concepts clear while the

Table 3. Program outcomes in the Course and corresponding Program outcomes description

Program outcomes	Program outcomes description	Cos addressed
1	Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization for the solution of complex engineering problems	1,4
2	Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences	3
14	Apply design and development principles in the construction of software systems of varying complexity	2

5th CO is intended for hands-on experience with components of an operating system. Table 3 lists the program outcomes addressed in the course.

Three programming assignments were designed, two from user level and one from Kernel level.

A. User level projects

I. Simulation of scheduling algorithms:Assignments can be given on the various scheduling algorithms including: first-come first-served, priority-based scheduling, round robin, and shortest job first. Each student was asked to implement any two scheduling algorithms. This project is worth 5% of the grade.

II. Implementation of shell: Shell is a command line interpreter that acts as an interface to OS. Student was asked to implement a simple shell with at least 5 new commands. This assignment enables the student to understand the relationship between child and parent processes, shell variables, the steps needed to create a new process, and a user-input parsing. This assignment is worth 5% of the grade.

B. Kernel level projects

First part of assignment included activity on building and installing kernel. Second part of the assignment is to add "Hello world" system call. This enables student to learn about operating system structure and also illustrates the difference between ordinary library call and system call. Kernel projects are extremely difficult and can be time-consuming..Hence students were recommended to make a team of 4 students. This assignment is worth 10% of the grade. The class strength is 80

I. Possible system calls

1. System call to print Hello world.

2. Getting the information about descendant processes of a given process.

3. Getting the information about processes given Process id.

4. System call to assign or remove an expiration time associated with a file.

5. System call to install or remove a watch point associated with a file.

6. System call to monitory the network I/O activity of a process or of all processes.

Table 4. Sample shell commands implemented

Command	Description
checkfile	This command checks if a given file is present in the system or no.
terminalcnt	This command counts the number of active users logged on to the terminal.
measure	This command converts the given measure in meter to its equivalent measure in centimetre, kilometre, feet and inches.
encrypt	This command takes one argument as input, the file name and encrypts the content of the given file
decrypt	This command decrypts the encrypted file back to the original form.
find_day	find day for given date and year

7. System call to monitor bandwidth quota of a process or of all processes.

The student was given the choice of using either XINU or Linux for this assignment. Students were given the useful links for adding system call for both Linux as well as XINU OS. For the implementation of shell focus was on novelty of commands.

Table 4 shows Sample shell commands.

Table 5. Assessment strategy

Week	Material	Weight age	
4	Simulation of scheduling algorithms	10M	Theory
6	Implementation of shell	10 M	Self study component
10	Adding system call	15 M	

Table 5 shows the assessment strategy employed.

Table 6 shows evaluation parameters.

6. Results And Discussion

Figure 1 shows the class average (in the scale 1-10) for the three assignments.

- The class score for the System call addition is slightly less than the other assignments. This assignment requires the student to learn how to use basic kernel data structures, how to copy data back from kernel space to user-space. Students were hesitant to try different system calls; there was repetition of system calls.
- Students were good in designing a shell. They came out with very good ideas in implementing new commands.
- Class score for the simulation of scheduling algorithm is better than system call, since students were through with the concepts by solving the scheduling problems.
- The present performance cannot be compared with previous year results since previous OS course was 5 credit course with self study component,

Students were asked to give their overall experience in performing this activity, Table 7 shows sample opinion.

Table 6. Evaluation Parameters

Simulation of scheduling algorithms		
• Assessment parameters	Allocated Marks	Remarks
• Implementation	6	
• Viva	4	
Implementation of shell		
Assessment parameters	Allocated Marks	Remarks
• Novelty and number of • commands	4M	
• Coding std and Data structures and	2M	
• Implementation	3M	
• viva	1M	
Adding system call		
• Assessment parameters	Allocated Marks	Remarks
• Novelty and usefulness of • call	4M	
• Implementation	9M	
• viva	2M	

Figure 1 shows average score of class in each assignment. Adding a system call activity addresses course outcome 5.

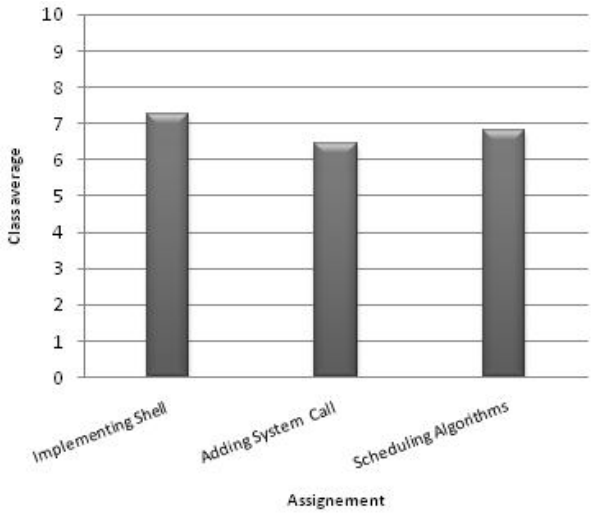

Figure1.Average scores of students in assignment

A. Students were able to understand operating system structure, to modify a large code base and manage its complexity. Student learnt how to understand the OS code written by others, use the existing code to extend its functionality by adding kernel level and user level applications. Since Linux is a production level OS, there is no much scope for addition of new system calls. Student who chose XINU OS for the activity had much wider scope in both second and third assignment. As this is our first experience in Kernel programming, we could explore

only technicality of addition of system calls. In future we intend of exploring other Operating systems so as to get hands on experience on various components.

Table 7. Feedback from students

XINU is quite interest ing, incomplete OS. It provides a lot of scope for kernel level projects.
The activity was quite interesting, we understood a lot about system call implementation, and the activity helped us realize the features provided by Linux, the importance of loadable kernel modules, and implementation of system call.
The activity was very helpful for understanding the structure of OS and understanding the technicality of adding system call

7. Conclusion

This paper presents the project based teaching of operating system. The authors used XINU as well as LINUX in these activities. XINU is a pedagogical OS where as LINUX is a real operating system. Two user level projects of moderate complexity and one kernel level project is designed as a part of CIE. It provided the students an opportunity to build kernel programming skills. This activity made the students to gain the knowledge of complete architecture of OS and also good C programming skills. The objective of these activities, to gain insight into technicality of adding system call has been attained. Next semester authors would pursue some more kernel level assignments.

References

[1] Giraldeau, Francis, and Michel R. Dagenais. "Teaching Operating Systems Concepts with Execution Visualization." 24 (2014): 1

[2] Nieh, Jason, and Chris Vaill. "Experiences teaching operating systems using virtual platforms and linux." ACM SIGCSE Bulletin. Vol. 37. No. 1. ACM, 2005.

[3] Laadan, Oren, Jason Nieh, and Nicolas Viennot. "Structured linux kernel projects for teaching operating systems concepts." Proceedings of the 42nd ACM technical symposium on Computer science education. ACM, 2011.

[4] Anderson, Charles L., and Minh Nguyen. "A survey of contemporary instructional operating systems for use in undergraduate courses." Journal of Computing Sciences in Colleges 21.1 (2005): 183-190.

[5] Aviv, Adam J., et al. "Experiences in teaching an educational user-level operating systems implementation project." ACM SIGOPS Operating Systems Review 46.2 (2012): 80-86.

[6] www.cs.tufts.edu/comp/150LNX/project/topics.html