Course Description

By itself, computer hardware is a lump of so much metal, plastic, and silicon. An operating system is a set of programs that turn this “bare” hardware into a useful computer system. The operating system’s primary goal is providing and managing shared access to the resources of the system. These resources include the CPU, memory, disks, terminals or displays, printers, and other peripherals. Operating systems range in complexity from simple systems for single-user personal devices through large multi-user systems to distributed systems managing cooperation among many connected computers. Real-time operating systems are specially designed for computing tasks where timely response to real events is vital; an example is the operating system used in the Mars rovers, which is based on a commercial off-the-shelf RTOS. One of the important tasks of an operating system is creating various levels of abstraction to hide the details of the structure and operation of the various hardware components that make up the computer, thus making the job of the applications programmer simpler.

The design of an operating system is an exercise in compromise rivaling the most delicate diplomatic negotiation. For any parameter of system performance that you seek to optimize in a design, there is a penalty to be paid in some other aspect of system performance. We will consider many of the tradeoffs involved in OS design and investigate many alternative methods of dealing with them. The course is arranged as a survey of these general design problems, but will also include case studies of how real systems have solved them.

In the programming assignments this semester, you will be implementing some components of an OS using the NachOS operating system simulator and the iPodLinux OS.

Prerequisites

CMSC 222, CMSC 301

Readings


In addition, the following books (among others) are available at the library for further reference:

- Unix Internals: the new frontiers, Vahalia
- Linux Kernel Development, Love (electronic version available via NetLibrary)
- Design and Implementation of the 4.3BSD Unix Operating System, Leffler, McKusick, Karels, and Quarterman.
- Design of the Unix Operating System, Bach. (Describes System V.)

Readings from the original literature will be assigned when appropriate, especially about the iPodLinux OS.
Web links

http://www.mathcs.richmond.edu/~lbarnett/cs321/
This URL gives access to general course information, including links to assignments, old tests, example code, and links to other web sites of interest. I’ll call this the “class web page.”

http://www.ipodlinux.org/
Home page for the iPodLinux project.

Grading Policy

Homework/Quizzes 15%
Projects 45%
Midterm 20%
Final Exam (non-cumulative) 20%

Assignments

There will be two types of outside-of-class assignments this semester, homework problem sets and programming projects. I encourage you to work together on homework assignments, but the writeup you turn in must be in your own words. There will be one midterm exam and a final exam.

The level of acceptable cooperation on the projects will be specified in the project assignments. The first project will be an individual project, and the others will be team projects.

The bulk of the projects will involve implementing pieces of the NachOS operating system. NachOS is an OS that runs as a single process on some other host operating system, in our case the Linux machines. It provides a basic OS functionality; you will be adding modules to flesh it out. NachOS is written in C++. You will be working in groups of two or three on these projects. Why use a simulated OS? First, it provides a safe, stable environment for your work which can always easily be restored to a known configuration. Changes to the OS can easily transform a working computer into a very sleek and expensive doorstop. It can require a great deal of time and effort to reverse the situation. Should you run into this sort of problem, it is much quicker and easier to recover from in a simulated environment.

Each member of the class will be loaned a fifth-generation iPod for use in this course. Our last assignment will involve installing iPodLinux on the iPod and modifying the OS to add some functionality. This will allow you to extend your experience from the safer NachOS environment to a real operating system running on real hardware.

I expect assignments to be turned in on time. For homework and individual programming assignments, a late penalty of 20% per day late will be assessed. Please be careful about picking up the wrong notebook on the way to class or otherwise leaving assignments where they are not accessible to you when they are due. There will be some built-in leeway for the group projects, which will be described in the first group assignment.

Writing expectations

I will evaluate your written work based on the “Other Student Criterion.” This means that you should provide enough written explanation so that another student in the class could read the submitted material and, without asking questions, understand the correct answer to the question. This applies to homework, quizzes, tests, and project write-ups. Basically, don’t write assuming that I am your audience, write assuming your peers are your audience. Don’t make me guess what you meant.
Honor System and Collaboration Policy

I assume that each of you is honorable. Tests and exams must be completed without help from others. Homework has been discussed above. For group projects, you are free to discuss whatever you like within your group. Discussion between groups should abide by the following collaboration rules:

- You may discuss compiler errors and issues relating to the computing environment with your classmates. It is acceptable to discuss the assignment with classmates, but not to use code written by another group in your solution to a problem. I will be happy to discuss problems in the logic of your program with you. For the first individual project, consider yourself to be a group of size 1.

Attendance Policy

Each unexcused absence will result in a 1 point deduction from your final average. Excuses need not be written, you just need to discuss the situation with me. When possible, please inform me of absences beforehand. For students who have five or more unexcused absences, I reserve the right to submit a grade of V, failure due to excessive absences.

Course Outline

In the following list of topics, the numbers in parentheses indicate sections in the text book which cover the topic. We will also cover material on the NachOS and iPodLinux systems as needed.

- Introduction to Operating Systems (Chapters 1, 2)
- Introduction to Processes and Threads (3, 4)
- Process scheduling (5)
- Concurrency and synchronization (6)
- Deadlock (7)
- Memory Management, Swapping (8)
- Virtual Memory (9)
- File System Overview (10)
- File System Implementation (11)
- Input/Output Hardware and Software (13)
- Disk systems (12)
- Protection and security (14, 15)
- Case studies (21, 22)

Test dates

Midterm: Friday, March 20

Final exam: Saturday, May 2, 7:00 – 10:pm