Advanced Operating Systems

Spring 2000
CS 63201 / 73201

Instructor

Dr. Robert A. Walker
walker@mcs.kent.edu
www.mcs.kent.edu/~walker

MSB 351, 672-4004 ext. 351
Office hours = Tu 1-3pm, Th 1-3pm, or by appt.

Teaching Assistant

to be determined…

Course Prerequisites

The *1999-2000 Graduate Schools Catalog* lists the prerequisites for this course as *CS 4/53201 Operating Systems*; equivalent courses taken elsewhere are also acceptable. It would be helpful if you are familiar with C++, as you will be writing and modifying C++ code in each of the projects, but if you are not, the subset that we will be using is easy to learn (although you might want to start learning it well before the first project is handed out).

Course Overview

The goal of this course is to provide an introduction to distributed operating systems. The first third of the course emphasizes *communication methods* that the OS must provide in a distributed system. The second third of the course considers how OS support for *synchronization and mutual exclusion* — central concepts in any operating system — must change in a distributed system. The final third of the course examines additional OS support required for a practical distributed system.

Note that this is a course in distributed operating systems, not a course in distributed computing or algorithms. Students interested in distributed computing (architecture and programming) should take **CS 6/73995 ST: Parallel and Distributed Computing** in the Fall 2000 semester. A course on distributed algorithms has occasionally been offered in the past, although there are no plans to offer it in the near future; a related course that might be of interest is **CS 6/76105 Parallel Algorithms**, offered each Spring Semester.

Textbook

The required textbook for this course is:

- *Distributed Operating Systems*, Sinha, IEEE Computer Society Press, 1997.

Other reasonable textbooks that you might want to refer to, if you have access to them, are:

- *Advanced Concepts in Operating Systems*, Singhal and Shivaratri, McGraw-Hill, 1994.
- *Distributed Operating Systems*, Tanenbaum, Prentice Hall, 1995.

Class Web Page

The web page for this class is **http://www.mcs.kent.edu/~walker/classes/aos.s00** (links to this page, and to my other classes, are all available on my home page). The web page will contain links to the following course materials:

- Current class syllabus and schedule
- Lecture notes (in PostScript and Adobe PDF format, printed 4-up)
- Programming projects, along with information about the Nachos instructional operating system
- Exam solutions

Other information may be included as well. You might want to check the web page on a regular basis, in particular when a programming project is outstanding.

Lectures

Students are expected to attend each lecture. I will not take roll, and I understand that it may occasionally be necessary to miss a class, but in general I expect you to attend each lecture.

At each class, I will hand out one sheet of paper containing reduced copies of *at most eight* of my slides for that lecture. If you would like to have reduced copies of *all* of my slides for that lecture, the full version of the lecture notes will be on the class web page before the lecture, and you can print them out. Note that you are not required to either look at or print out these notes; they are provided solely for your convenience should you want them. However, you should ***not*** consider skimming these notes to be an adequate substitute for attending the lecture, as they will contain only the text of my slides, not the comments that I will make in class.

My lecture notes will be drawn from a variety of sources. The required text will serve as a primary reference, although some material will be drawn from other books on operating systems and distributed operating systems. In the past, I have used lecture notes from other professors as a reference, in particular notes by Jorg Liebeherr, Steve Chapin, Thomas Doeppner, and Kathryn McKinley. I plan to also use a couple of lectures developed by Prof. Mikhail Nesterenko this semester.

Programming Projects

There will be two programming projects during the semester, both based on the Nachos instructional operating system, and involving reading and writing code. Tentative due dates are shown on the Class Schedule, attached at the end of this syllabus.

Nachos Programming Projects

The Nachos instructional operating system is written in C++ (actually, a subset of C++ that uses classes and methods, but avoids troublesome C++ constructs like inheritance and overloading). If you need quick refresher on C++, see the document “A Quick Introduction to C++” on the class web page.

Late Policy

In general, you will have adequate time to complete each assignment. However, you should begin work on each assignment early so that you will have plenty of time to become familiar with it and with the Nachos code that you must read and/or modify, and so that you will have time to “sleep on” the difficult parts. Waiting until two days before the due date to start the project is a bad idea.

For programming projects, late projects ***will*** be accepted with a 10% penalty for ***each day or portion thereof*** that the project is late. Other extensions will not be granted, unless you make *prior* arrangements with me, or have a *documented* illness (in which case I expect you to contact me as soon as possible).

Exams

There will be two exams (held during class) and a final exam (held during finals week). The tentative dates for the exams are shown on the Class Schedule, attached at the end of this syllabus. All exams are closed book and closed notes, and must be individual work. It is expected that you take each exam at the scheduled time, unless you make *prior* arrangements with me, or have a *documented* illness (in which case I expect you to contact me as soon as possible).

Academic Integrity

Student-teacher relationships are built on trust. Students must trust that teachers have made appropriate decisions about the structure and content of the courses they teach, and teachers must trust that the assignments which students turn in are their own. Acts which violate this trust undermine the educational process. In this course, the penalty for ***any*** act of academic dishonesty is a final course grade of F.

Cooperation on Programming Projects

For programming projects, I strongly believe that discussion with your peers is an excellent way to learn. If you don't understand something, discussing it with someone who does can be far more productive than beating your head against the wall.

Having advocated discussion, then, I must be about clear what is allowed, and what is not. In general, students are allowed to cooperate as follows: you are allowed to discuss with other students *the assignment*, and *general methods for solving the assignment*. However, you are ***not allowed*** to work with someone else to actually *solve* the assignment, or to *write code* (even pseudocode) for a program, and you are certainly ***not allowed*** to *copy* anyone else's solution; doing any of these things will be considered cheating, and will be grounds for failing the course.

Note that there is a fine line between discussion and cheating. If you are unsure what is allowed and what isn't, feel free to discuss the distinction with me, but if something feels uncomfortable, it's probably not allowed.

Finally, you should be careful not to give others access to your code. This means that you shouldn't keep your program in a publicly-accessible directory, you shouldn't leave your terminal unattended, and you shouldn't forget to pick up your printouts.

Grades

Your final course grade will be broken down as follows:

• Programming projects (2)	40%
• Exams (2)	30%
• Final exam	30%

The final course grade will be determined with A = 90–100, B = 80–99.99, etc. There will be no curve at the end of the course, although individual exams, projects, etc. may occasionally (although rarely) be curved. Thus you should always be able to determine how well you are doing in the course.

Students With Disabilities

In accordance with University policy, if you have a documented disability and require accommodations to obtain equal access in this course, please contact the instructor at the beginning of the semester or when given an assignment for which an accommodation is required. Students with disabilities must verify their eligibility through the Office of Student Disability Services (SDS) in the Michael Schwartz Student Services Center (672-3391).