CPS 110: Operating Systems

Course Overview
Fall 1997

Instructor: Jeff Chase (*chase@cs.duke.edu*)
Class Meetings: Tuesday and Thursday, 9:10 - 10:25 (D106 LSRC)
Teaching Assistant: Darrell Anderson (*anderson@cs.duke.edu*)

This is an introductory undergraduate course in operating systems, emphasizing the core operating system concepts of protected kernels, processes and threads, concurrency and synchronizations, file systems, and virtual memory. This semester will also include a fleeting introduction to distributed systems concepts. A major component of the course is a series of projects using the notorious Nachos instructional operating system developed by Tom Anderson at Berkeley.

Who should take this course. All undergraduate CS majors should take this course at some point. You should have completed the basic introductory sequence, and also CPS 104. Familiarity with C, C++, and basic Unix program development tools (e.g., *gmake*) is assumed.

Readings and resources. The primary readings are from the book *Operating Systems Design and Implementation* by Andy Tanenbaum. This book has a lot of heavy, gritty, and expensive detail on Minix, a system we won't use or discuss much, but it is in other respects the best choice for this course. In an effort to balance the inevitable Unix slant, there will also be optional readings from papers and the book *Inside Windows NT* by Helen Custer. We won't be using Microsoft's NT in this course, but you should learn something about it, and Custer's book is a readable if slightly out-of-date treatment of this successful modern operating system.

All course handouts and supplemental materials are featured on the course web page at *http://www.cs.duke.edu/~chase/cps110*. There is also a course newsgroup *duke.cs.cps110*. Darrell and I will read the newsgroup regularly. We prefer that you post questions to the newsgroup rather than e-mailing us. If you know the answer to a posted question, please post a response.

Workload. This is a demanding course. There are seven required Nachos programming assignments, with one due every two weeks. We may add two optional Nachos projects (on file systems and networking) for extra credit in the second half of the semester. To make things a bit easier, you will work through the Nachos projects in teams of two or three students. As in previous offerings of CPS 110, you will present your projects in demo sessions with me or with Darrell, with a different "spokesperson" for your team each week. These projects will be fun and valuable if you form teams quickly, start early on the projects, and make your Nachos work a priority this semester. You will have at least one and perhaps two opportunities to change teams, so your initial selection is not critical.

In addition to the Nachos assignments, there will be optional problem sets and two in-class closed-book exams. Expect that the second of these exams will be a scheduled final after classes end.

Grading. The Nachos projects are worth 50% of your grade. Your exam scores are counted as 45% of your grade. The problem sets are ungraded, but about half of the exam questions are drawn from the problem sets. The remaining 5% of your grade is based on "subjective factors" including class participation, reports from your peers (see below), and your performance on any random quizzes that I may decide to give (possibly unannounced).

I grade on a curve to compensate for any possible problems with the projects and exams. The center of the curve varies from semester to semester, but I always reward students who complete the projects and demonstrate mastery of the course material. I am not known for being a punishing grader, but don't rely on that: if you do not take the projects seriously, then you will fail the course. Extensions will be rare and well-justified.

Topics. The course covers five primary topic areas.

- *Concepts in operating systems.* Operating system services, operating system structure, processors and processes, protected kernels, traps, interrupts, processes, and threads.
- *Concurrency, synchronization, and communication.* Threads, mutual exclusion, events, scheduling, and deadlock. Implementing and using synchronization.
- *File Systems.* The storage hierarchy. File and directory structures, file cache management, dealing with disks, failure and recovery, access control and security.
- *Virtual Memory.* Address spaces, protection, virtual address translation, control of memory management hardware, and physical memory management.
- *Networking and distributed systems.* Network communication, consistency, and failures. Network RPC, network services, and the client/server and peer-peer models. Problems and challenges in distributed computing. Distributed file systems and memory management for workstation clusters.

Policy on collaboration for CPS 110. I encourage collaboration on problem sets and, of course, the Nachos projects. Help each other. To encourage cooperation, we will award "brownie points" that improve your grade in the "subjective factors" category if your peers report that you have helped them. Of course, you also have a responsibility to acknowledge anyone who has helped you with your work. Failure to adequately acknowledge your contributors is at best a lapse of professional etiquette, and at worst it is plagiarism. Plagiarism, like collaboration on exams, is cheating. Cheating is a very serious offense and I do not expect it to occur in this class. Written work turned in as your own must be your own, and I may call upon you to explain (alone) your choices and answers in more detail.