CMSI 387
OPERATING SYSTEMS

http://myweb.lmu.edu/dondi/spring2010/cmsi387

Spring 2010 — Doolan 222
TR 10:50am–12:05pm, 3 semester hours
Office Hours: TR 3–6pm, or by appointment

John David N. Dionisio, PhD
e-mail: *dondi@lmu.edu,* AIM: *dondi2LMU*
Doolan 106; (310) 338-5782

Objectives and Outcomes

The primary objective of this course is to master the fundamental concepts behind modern operating systems, with some comparative study of real-world systems. Understanding conceptual issues and mechanisms on their own, without confusing them with a particular operating system's specific policy, implementation, or interface, is crucial to being able to learn, use, and control any system effectively and quickly. This understanding should also enable you to learn new operating systems more easily, as they ultimately build upon the same principles and concepts that will be examined in this course.

Prerequisites/Prior Background

Programming proficiency in a systems-level language, particularly C; a prior course in computer system organization (LMU CMSI 284 or equivalent). Familiarity with Java, shell scripting, and system administration is also beneficial.

Materials and Texts

- Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne, *Operating System Concepts*, 8th edition, John Wiley & Sons, 2008 — *a.k.a. "SGG"*
- Assorted handouts, articles, and sample code to be distributed throughout the semester

The following texts are recommended and not required — but they *will* fill in a lot of specifics:

- Daniel Pierre Bovet and Marco Cesati, *Understanding the Linux Kernel*, Third Edition, O'Reilly Media, 2005
- Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman, *Linux Device Drivers*, Third Edition, O'Reilly Media, 2005

Additional information is also available on the web; do not hesitate to look for further sources of information regarding the concepts, techniques, tools, and paradigms that we will discuss.

Course Work and Grading

Graded coursework consists of homework (25%), 1 midterm (25%), 1 kernel project (25%) and 1 final exam (25%). Letter grades are determined as follows: ≥ 90% gets an A– or better; ≥ 80% gets a B– or better; ≥ 70% gets a C– or better. The instructor may curve grades upward based on qualitative considerations such as degree of difficulty, effort, class participation, time constraints, and overall attitude throughout the course. Grades are never curved downward.

Homework

Homework consists of questions, exercises, and programming assignments to be given throughout the semester. Homework is where you can learn from your mistakes without grading penalty: if you do the work and submit it on time, you will get full credit, regardless of correctness. What goes around comes around: the effort you put into your homework pays off in the tests and the kernel project. The homework submission deadline is always the beginning of class on the designated due date; the due date is encoded in the homework number. Submissions after the deadline receive half credit, period. Extra credit homework may be assigned; fulfilling this is counted on top of the 25% allocation of homework to your final grade.

Tests

The midterm is scheduled for March 2; the final exam is scheduled for May 4. The tests are meant to assess the foundational knowledge presented in the course; questions include content-oriented elements as well as forward-looking, applicative portions (i.e., "use this knowledge to resolve this situation"). Tests are open-paper-everything; no sharing. "Open computer" might be allowed depending on the circumstances. You may neither solicit nor give help while the exam is in progress. Late and/or missed tests are handled on a case-to-case basis; in all instances, talk to me.

Kernel Project

The general principles behind the material will find specific application in an operating system kernel project. The project may either be a demonstrable, nontrivial modification or enhancement to an operating system kernel, or a program or driver that interacts with the kernel in an interesting and significant manner. As such, Linux is the most likely target technology due to the availability of its kernel's source code; projects involving other computing platforms, especially embedded, mobile, or gaming devices, are also fair game, as long as sufficient technical resources are available for them.

The kernel project will be graded according to the following criteria:

1. *Design (30%):* How good is the overall structure of the code? Is it clear, flexible, and easy to maintain? Is it elegant or innovative? How well does it apply the principles of "separation of concerns" and "one change, one place?"
2. *Functionality (30%):* How well does the code work? Does it fulfill requirements? Are its results accurate or correct? Does it perform its tasks in a reasonable amount of time? How well do unit tests validate the code?
3. *Naming (20%):* Are program entities — classes, subroutines, variables, etc. — clearly and consistently named? Do their names correspond to their functions and roles?
4. *Comments (15%):* Are comments provided where appropriate? Are they clear and well-written? Does the code take advantage of any special support for comments provided by the project language or platform?
5. *Version control (5%):* Is the code committed at reasonable intervals? Are milestones appropriately tagged? Are adequate descriptions of provided in the commit logs?

Kernel project deliverables are due on May 4. Late projects will not be accepted.

Attendance

Attendance at all sessions is expected, but not absolutely required. If you must miss class, it is your responsibility to keep up with the course work. Note that the last day to add or drop a class without a grade of W is January 25. The withdrawal or credit/no-credit status deadline is March 26.

Special Accommodations

Students with special needs who need reasonable modifications, special assistance, or accommodations in this course (such as a documented disability [physical, learning, or psychological]) should contact the Disability Services Office (Daum Hall, Room 224, x84535, *http://www.lmu.edu/dss*) as early in the semester as possible. All discussions will remain confidential. In addition, please schedule an appointment with the instructor early in the semester to discuss any accommodations for this course for which you have been approved.

University Policy on Academic Honesty

Loyola Marymount University expects high standards of honesty and integrity from all members of its community. All students are expected to follow the LMU honor code, as stated in the *LMU Undergraduate Bulletin 2008-2010*, pp. 58–59 (online at *http://www.lmu.edu/Page13245.aspx*).

Topics and Important Dates

This schedule may change based on the actual ebb and flow of the class; deadlines, exams, and university dates (italicized) are less likely to change than lecture topics.

January	Operating systems overview; operating system "power use"
January 25	*Last day to add or drop a class without a grade of W*
February	Booting and installation; process management
March	Scheduling and synchronization; memory management
March 2	Midterm
March 26	*Withdraw/credit/no-credit deadline*
March 29 to April 2	*Spring break; no class*
April	Mass storage and file systems; additional topics (time permitting)
May 4	*Final exam, 11am; kernel projects due*

You can view the class calendar on the web at *http://ical.me.com/dondi/LMU*, or via iCalendar at *webcal://ical.me.com/dondi/LMU.ics*.