ECSE 427/COMP 310 -- Operating Systems
January 2007

General Information

Instructor:	Muthucumaru Maheswaran
Tel:	398-1465
Email:	**maheswar@cs.mcgill.ca**
Office:	Room 213B, McConnell Engineering Building
Office hours:	TBA. Appointments can be made for meetings at other times.
Email:	Please use SOCS email for urgent messages. For other messages (particularly with large attachments) use WebCT mail.
Class:	ENGTR 0100
Tutorial:	TBA
Prerequisites:	see Calendar
Class web page:	**http://www.cs.mcgill.ca/~cs310**. WebCT will be used for assignments.
TAs:	TBA
TA office hours:	TBA

"McGill University values academic integrity. Therefore, all students must understand the meaning and consequences of cheating, plagiarism, and other academic offences under the Code of Student Conduct and Disciplinary Procedures (see www.mcgill.ca/integrity for more information)."

Course Description and Learning Outcomes

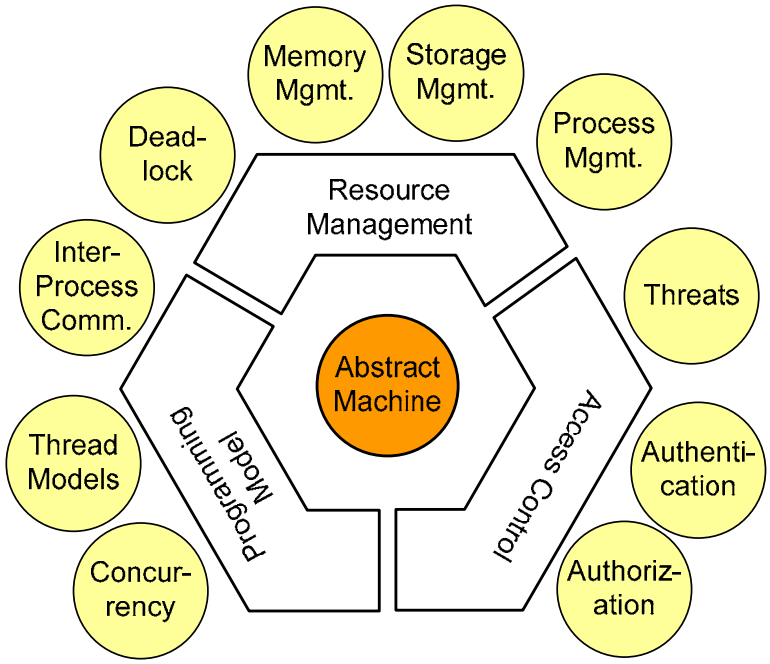
Description: This is an introductory course in computer operating systems. In this course we will study the theoretical and practical concepts behind modern operating systems. In particular, we will study the basic structure of an operating system, its components, design strategies, algorithms and schemes used to design and implement different components of an operating system. Major components to be studied include: processes, inter-process communication, scheduling, memory management, virtual memory, storage management, network management, and security.

Primary learning outcome: To get a clear ***understanding*** of the major principles/algorithms that underlie an operating system and how they interplay within it.

Secondary learning outcomes: After taking this course, you should be able to: (i) **identify** the core functions of operating systems and how they are architected to support these functions, (ii) **explain** the algorithms and principles on which the core functions are built on, (iii) **explain** the major performance issues with regard to each core function, and (iv) **discuss** the operating system features required for a particular target applications.

Course Content

CONCEPT MAP



Course Schedule

The table below shows a tentative course schedule (I will make a "best effort" to stick to the proposed schedule).

Date	Topics	Reading Material	Comments
Jan 3	Overview of the course, assessment details	Course outline	
Jan 5	A whirlwind tour of an OS: the big picture	Chapters 1, 2	WA #1 assigned
Jan 8	A whirlwind tour of an OS… (part 2)		
Jan 10	Storage management: concepts, issues, approaches;	Chapter 10 (minus 10.5.2-10.5.3)	
Jan 12	File systems: naming, attributes, directories, access, and location; File system implementation;	Sections 11.1-11.7	
Jan 15	File system implementation… (part 2)		PA #1 assigned
Jan 17	File system implementation… (part 3)		
Jan 19	File system implementation… (part 4)		

Jan 22	Disk scheduling	Sections 12.1-12.6	**Learning milestone #1**
Jan 24	Process management: definition, states, scheduling;	Sections 3.1-3.3	
Jan 26	Process scheduling… (part 2)	Chapter 5 (minus 5.4, 5.5, 5.6.1, 5.6.2, 5.7.2-4, 5.8)	
Jan 29	Synchronization: concurrency, mutual exclusion, examples, semaphores, monitors, more examples;	Chapter 6 (minus 6.8-6.10)	
Jan 31	Synchronization… (part 2)		
Feb 2	Synchronization… (part 3)		
Feb 5	Synchronization… (part 4)		
Feb 7	Synchronization… (part 5)		
Feb 9	Deadlocks: classification, prevention, avoidance, detection; Bankers algorithm; RAGs	Chapter 7	WA #2 assigned
Feb 12	Deadlock… (part 2)		
Feb 14	Inter-process communication; signals	Sections 3.4-3.6	**Learning milestone #2**
Feb 16	Basic networking: sockets	Handouts on networks	
Feb 19	####		
Feb 21	**Spring Break Week**		
Feb 23	####		
Feb 26	**In-class midterm**		
Feb 28	Networking… (part2)		PA #2 assigned
Mar 2	Executable file management: linking, loading, file placement in memory	Handouts (maybe)	
Mar 5	Memory management: storage allocation, garbage collection, segmentation, paging, locality, cache management;	Chapter 8	
Mar 7	Memory management… (part 2)		
Mar 9	Memory management… (part 3)		
Mar 12	Virtual memory: page replacement, placement, thrashing, working sets;	Chapter 9	
Mar 14	Virtual memory… (part 2)		**Learning milestone #3** -- PA #3 assigned
Mar 16	Protection problem: protection domains, access matrix, multi-level security,	Chapter 14	

	information flow;		
Mar 19	Protection… (part 2)		
Mar 21	Security: authentication, cryptography, authorization	Chapter 15	
Mar 23	Security… (part 2)		
Mar 26	Security… (part 3)		
Mar 28	Threats: program, system, and network threats, firewalling		
Mar 30	Threats… (part 2)		
Apr 2	Trusted computing	Handouts	WA #3
Apr 4	Trusted computing… (part 2)		
Apr 6	Trusted computing… (part 3)		
Apr 9	Class review		**Learning milestone #4**

The course will consist of three hours of instructor led classes per week together with a *maximum* of one hour of tutorial per week taken by the TAs. The class time will be devoted to the presentation and development of new concepts and the application of these concepts to examples and problems, while the tutorials will discuss solutions to the programming projects and written assignments. The primary focus of the tutorials is to provide sufficient "how-to" knowledge through the discussion of the assignments to help in the development of the programming project series.

Instructional Method

The course will consist of three hours of instructor led classes per week together with a *maximum* of one hour of tutorial per week taken by the TAs. The class time will be devoted to the presentation and development of new concepts and the application of these concepts to examples and problems, while the tutorials will discuss solutions to the programming assignments/projects and written assignments. The primary focus of the tutorials is to provide sufficient "how-to" knowledge through the discussion of the assignments to help in the development of the programming project series.

Students are strongly encouraged to use the WebCT discussion groups to talk about the programming and written assignments. These discussion groups will be monitored by the TAs and by the instructor for providing the necessary answers.

Course Materials

Textbook

Required textbook: A. Silberschatz, P. B. Galvin, and G. Gagne, *Operating System Concepts*, 7th Edition, John Wiley & Sons, 2005, Hoboken, New Jersey.

Evaluation

The following table shows the learning outcomes that should have been met at the different learning milestones. These learning outcomes reflect the knowledge you should have acquired through the learning process at that given point in the semester.

I have posed the learning outcomes as questions so that you can test your knowledge with this aspect of the operating system. The idea is that you should be able to comfortably answer different flavors of the question posed as part of the learning outcome.

Learning milestone	Outcomes to be met (should be able answer the following questions)
#1	Why do we need an OS? What are the core functions of an OS? In putting the core functions together, what architectural choices can we make? What trade-offs exist between the different architectural choices? What are the performance measures relevant in evaluating some OS functions? What are the key operations performed by a storage manager? What are the key issues in storage management? What is an important realization of storage management? Describe the anatomy of a simplified file system? Can you implement a simplified file system using this knowledge? Why do we need to disk scheduling? What are the possible approaches for disk scheduling?
#2	What is process management? How is process management done in UNIX? What are the possible approaches to process scheduling? What is process synchronization? Explain why process synchronization is key consideration in an OS? What is a deadlock? What are the approaches for managing deadlock? Can you build operating systems that do not specifically deal with deadlocks? What can happen with such operating systems? What are the approaches for inter-process communication?
#3	What support is provided by OS for networking? What are the mechanisms provided by the socket API of UNIX? What are the processes taking place in linking and loading object files? Why is memory management important? How does memory management support multiprogramming? Do we need memory management in simple uniprogramming machines?

	Why is virtual memory necessary? Describe important issues with regard to virtual memory? Does virtual memory always improve the performance?
#4	What is the protection problem? Why is it important in multi-user system? Is protection important in single-user systems? What is the basic security problem in operating systems? List important threats with regard to operating systems? What are the possible counter measures? What are the ideas behind trusted computing?

Activity	**Percentage**
Written Assignment 1: Simple OS questions	3%
Written Assignment 2: Synchronization & deadlocks	3%
Written Assignment 3: Protection & trusted comp.	4%
Programming Assignment 1: Disk simulation	5%
Programming Assignment 2: Simple file system	10%
Programming Assignment 3: Simple networked file system	15%
Midterm	10%
Final (comprehensive)	50%
Total	**100%**

Written Assignment #1: Simple OS questions
This assignment is meant to cover some of the computer architecture concepts you studied in the prerequisites and start you thinking on some basic OS problems. It can involve some basic scripting questions or operating system commands. You need to run the OS commands and report the behavior in the assignment.

Written Assignment #2: Synchronization & deadlocks
Synchronization and deadlocks are two theoretical concepts we will cover in this course. This assignment is meant to give you some practice in solving some interesting problems in these topics. You will write down solutions to well known synchronization problems.

Written Assignment #3: Protection and trusted computing
This assignment will give you an opportunity to get familiar with notions in OS protection and security. No programming involved here.

NOTE: The official programming language of this course is C. This means all support for programming assignments will only be provided in C. However, you can handover programming assignments in Java as well – you are responsible for solving Java issues.

Programming Assignment #1: Disk simulation
A basic disk simulator will be give to you. It is written in C. If you are doing your assignments in Java, you could convert this to Java. This assignment requests you to add some functionality such as error rates, delays, actual data reads and writes to the given simulator.

Programming Assignment #2: Simple file system
You are expected to implement a simple file system. Detailed description will be given. You are expected to merge the parts completed in #1 with this one.

Programming Assignment #3: Simple networked file system
You are expected to implement a simple networked file system. Detailed description will be given. You are expected to merge the parts completed in #1 and #2 with this one.

Cumulative Grading Policy: In this course, we will use a cumulative grading policy. It will consider programming assignments and written work separately. In the programming assignments, PA #3 can replace PA#1 and PA#2 if they are less. Similarly, the Final exam can replace other written assignments if they are less.

Late Assignment Policy: There will be two deadlines for each assignment: proper deadline and cut-off date. After the proper deadline, there will be a penalty of 10% for each day the assignment is late until the cut-off date. After the cut-off date, the assignment cannot be handed in. No individual requests for extensions will be granted unless they are for medical reasons.

The deadlines will be set for 11:55 pm or 11:59pm. Please observe the time and date very carefully. It is your responsibility to make sure that the assignment is properly submitted via the WebCT.

Regrading Policy: If you find your assignments or exams are not marked according to the marking scheme, you are encouraged to consult me or the TAs. When you resubmit your assignment or exam for regarding, we reserve the right to regrade the full exam or assignment without restricting the attention to the disputed portion.