CS 3204 Syllabus

Overview

CS 3204 provides an in-depth introduction to the principles and practices of operating systems. Particular emphasis is given to the topics of multiprogramming, process and thread management, memory management, including virtual memory, concurrency, including synchronization and deadlock, resource allocation and management, including scheduling, and storage management and file systems. Additional topics include inter-process communication, networking, and device management.

Rather than learning what an OS looks like from the outside and how to use its facilities, this course will show you what an operating system looks like from the inside.

The topics will be accompanied by a series of programming projects that will give you hands-on experience in building significant parts of a real operating system. Most projects will be done in groups, which gives you the added benefit of learning how to work in a team.

Staff Information and Meeting Times

Instructor: Dr Godmar Back

VTKW-2211

1-3046

Office hours: Tuesday 8:00 am – 9:15 am in MCB 637.

Thursday 1:30 pm – 3:30 pm in MCB 637.

On Thursday, I may be leaving early if there is no traffic, but I will stay until at least 2:30pm.

I'm also available by appointment. Check my Google calendar before asking for an appointment.

Class website: http://courses.cs.vt.edu/~cs3204/fall2008/gback

GTA: Due to the budget limitations resulting from reduced enrollment, we are operating with limited GTA support this semester.

Xiaomo Liu (50%)

Peter Radics (100%)

Office hours will be held in MCB 106*. Please see website for office hours. Additional office hours will be announced in the forum when projects are due.

The GTAs are also available by appointment.

Email: To contact teaching staff, use cs3204-staff@cs.vt.edu

Forum: http://forum.cs.vt.edu/

Class Meeting Times:

Section 1: MCB 231 9:30am - 10:45am T R

Section 2: MCB 230 11:00am - 12:15pm T R

Regular class attendance is not enforced, but is strongly recommended. Subjects taught in class closely correspond to the concurrently run projects.

Prerequisites

The formal prerequisites for this class consist of CS2604: Data Structures and File Processing and ECE/CS 2504: Intro to Computer Organization. Each student must prove that they have obtained a grade of C or better in those classes. Students who fail to meet the prerequisite requirements should drop the class. Students who meet the requirements must submit a Computer Science Prerequisite Form as soon as possible, but no later than by the end of class on Sep 2, 2008 for verification.

The most important informal prerequisite consists of strong C/C++ programming skills. In particular, the projects in this class will require a solid understanding of pointer-based data structures.

A user-level understanding of the Linux operating system is required as well, as would be provided in CS 2204: UNIX.

Objectives

Upon completion of the course, students should be able to

1. Understand the basic structure and organization of a multi-programmed computer system, including the distinction between user and kernel mode, the use of interrupts and context switches, runtime organization, application-binary interfaces and system calls, program linking and loading.
2. Understand the principles underlying concurrency and know how to use proper synchronization and deadlock avoidance techniques.

3. Understand the principles behind memory management, including user-level memory management, virtual memory management and paging.
4. Understand the principles behind CPU scheduling, including round-robin, priority-based, and multi-level feedback queue based scheduling algorithms.
5. Understand how an OS provides protection to its applications and how it manages and virtualizes resources.
6. Understand how file and storage systems are constructed and what factors influence their performance.

At a higher level, I would like for you to take away an appreciation for the complexity of operating systems, and view this class as an example of how to learn managing complexity.

Textbook

The required textbook is:

Silberschatz, Galvin, and Gagne. *Operating System Concepts*, 8th Edition, John Wiley & Sons, 2008. ISBN 0-470-12872-0.

I will post lecture notes after each lecture on the class website. In addition, the class website will link to other external resources.

Format

The course work consists of a mix of lectures, exams and programming projects.

Midterm: There will be one in-class midterm. The midterm will cover material from the lectures and textbook. The midterm will also include questions related to the programming projects.

Final: There will be a final exam. The final exam will be comprehensive and include material from the lectures, textbook, and programming projects.

Projects: There will be 5 projects. You will do the first project individually; the remaining four projects will be done in a group, except as noted in the project description. Some of the group projects will have an individual component.

Projects will be submitted electronically and grades will be posted electronically. Instructions will be posted on the class website.

Projects

For the sixth semester now we are using Pintos, an educational operating system developed by Ben Pfaff at Stanford University. Pintos is used at a number of universities in comparable undergraduate OS courses. We provide a baseline version of Pintos, and you will add various features to Pintos through the course of the semester. Pintos runs on the x86, and could boot on

any PC, but we will use a virtual machine simulator to run it on the Linux remote login cluster. (We do not provide support to run Pintos anywhere else and we will grade your submissions on those machines.)

The CS 3204 projects are probably different from the projects you have encountered in the undergraduate classes you took so far.

First, they are probably harder than what you have done so far. Although a correct solution can be implemented in a total of about 2,500 lines of C code, getting your projects to actually pass the tests will require intense coding and debugging. We will provide you with tools and help.

Second, unlike most assignments you've done so far, you will work in a group. Working in a group more closely resembles what you will encounter outside academia, but it requires trust and cooperation among group members. Note that your grade will be determined by the performance of your group. All group members will receive the same score for a project, unless otherwise specified. Do not expect to farm out work and receive credit for a partial solution. For more details, read the collaboration policy outlined later in this document.

Third, you will have to read a substantial amount of code *before* you can start writing the first line of code in your solution. We will guide you to the parts of the code you have to study, but do not expect to start coding right away. We believe this to be beneficial in two ways: first, this mirrors what you encounter outside academia. Second, we believe you will pick up good programming practices by reading well-written and well-documented code.

Fourth, only 50% of your grade in those projects is determined by passing the test cases we provide (all of which are public). The other 50% of your score comes from the design documents you are required to submit with your solution. This mirrors an industrial setting where you have to provide a rationale for your design and create documentation that makes your code maintainable. In the past, some student felt that assigning "only" 50% to the test did not reflect the proportion of work put into them. Note, however, that passing the tests is a precondition for receiving full credit for documentation – we don't give credit for documentation when a test shows that a solution you describe does not actually work.

Fifth, you will be using the C language, rather than C++, for these projects. This mirrors most current OS, which are written in C rather than C++. Using C instead of C++ means that you cannot use certain language constructs, such as classes or templates, or runtime features such as RTTI. Nevertheless, your code will be

written in an object-based style with encapsulation that is similar to how a C++ program would be structured.

The original Pintos series was designed for a 10-week quarter. This allows us to add a “warm-up” project at the beginning, and to relax the schedule for the projects somewhat. Nonetheless, you do not have the option to procrastinate.

We encourage you to start asking questions early and often.

Late Policy

I am generally hesitant to grant individual extensions for projects. Instead, each student will have a budget of 4 late days that can be used to submit projects late without penalty. You decide when you want to use your late days – there is no need to contact the instructor or GTA beforehand. Late days are granted in whole integer multiples of days: if your assignment is 5 minutes late, you will have used up an entire late day. *Submissions received after you have used up your late days will receive a zero score.* For some assignments, you may work in a team. If you are working in a team, late submissions will count against the budgets of all team members, so make sure that all of you have enough late days left or the team member with an insufficient number of days risks getting a zero.

These late days are intended to account for various minor emergencies, such as network outages, snow or flood days, or lab/cluster downtime: please contact the instructor for extensions only if you have truly extraordinary circumstances that would prevent you from completing the assignments on time. Job interviews do not count as extraordinary circumstances.

Late Drop and Incomplete Policy

Less-than-hoped-for performance, or realizing you have taken on too much work this semester, are not permissible reasons to grant course withdrawal requests after Oct 3, 2008. (This policy applies only to drop requests that have to be approved by the instructor; in particular, it does not apply to the course withdrawals for six credit hours to which you are entitled according to college policy.)

I will not grant incompletes for this course unless truly extraordinary, unforeseen circumstances outside of your control are to blame.

Grading

I estimate that the contributions of the different portions to your final grade will be as listed below, but I reserve the right to adjust these weights as necessary:

15%	Midterm Exam

30%	Final Exam
55%	Projects

I expect the median final grade for this class to be between a B and B-. In other words, students who consistently perform above the median can expect a B or better. The sample population over which this median is computed includes all students who submit at least one piece of work for grading during the course of the semester. We will publish score distributions for all projects and the midterm to give you an indication of where you are. I reserve the right to adjust this curve in either direction, depending on the performance of the class.

Students who do not appear for an exam at the scheduled time will receive a zero score on the exam.

Auto-Fail Rules

You fail the class, unless:

- You produce a working Project 2 by the end of the semester. "Working" means for your project deliverable to pass a to-be-defined set of tests.
- You show "reasonable effort" on Projects 3 and 4. A zero score on any of these two projects clearly fails the "reasonable effort" test – otherwise, we will use our judgment as to what constitutes reasonable effort.

Note that meeting these requirements forms a necessary, not a sufficient condition for passing. "Passing the class" means a grade that is not an F.

Modes of Communication

Website: The class website and the forum are the primary means of communication from us to you. We will post announcements there, which are binding. There is no mailing list for this class.

Email etiquette:

- Class-related email should be sent to cs3204-staff@cs.vt.edu, with the exception of project-related questions, which should be posted in the forum. The email alias ensures that it reaches both the instructor and TA, and ensures that it will be archived separately. I will answer questions posted in the forum before answering questions I receive by email.

- Please set your full name in your email client so it shows in the From: line in email you send. I may not respond to email sent from aawuhu24@vt.edu if I don't recognize the sender as a student. Email sent from certain accounts (e.g., hotmail) may end up in my spam folder and be overlooked. Do not send me attachments unless asked to do so.

Collaboration Policy and Honor Code

On the class website you will find links to the following policies applying to this class: University Policy of Class Attendance, the The Virginia Tech Undergraduate Honor System, the ACM and IEEE Code of Ethics, and the Departmental Policy on Koofers.

The tenets of the honor code will be strictly enforced in this course, and all assignments shall be subject to the stipulations of the Undergraduate Honor Code. For more information on the Honor Code, please refer to the Undergraduate Honor System Constitution, located online at http://www.honorsystem.vt.edu/

In addition, we will enforce the Departmental Policy on Koofers, Old Programs, Cheating, and Computer Use, available at http://www.cs.vt.edu/undergraduates/handbook.html#Koofer

The following policies regarding collaboration apply in this class.

- All submitted work is expected to be the original work of the individual student unless otherwise directed by the instructor. Note the emphasis on "submitted" work – this includes work that is explicitly graded and work that may not be.
- Projects are to be the work of the individual student or team as specified. You may discuss general concepts, such as software libraries, Internet resources, or class and text topics, with others outside your team. However, discussion of project solutions, specific code, or detailed report content is an honor code violation. All source material used in project code and reports must be properly cited.
- For the projects except the first, you will team up in groups of 3 students. While teaming up is voluntary, we strongly recommend that you work with a full group. You may switch teams or form new teams, but only between projects. You may work with at most one group on a given project. Students must contribute equally to the project within a team. It is not acceptable for students to either not contribute to the project or not to let the other group members contribute equally to the project. Please bring any problems in this regard to the instructor's attention early on.

- You are required to read-protect your work on shared file space so students outside of your team will not have access. Failing to do so is an honor code violation.
- Borrowing code or hiring someone to perform the work for you is an egregious violation of the honor code. We will use plagiarism detection software such as MOSS to screen out students attempting to do this. We have access to project solutions that were created and submitted outside of Virginia Tech, as well as all solutions that were submitted in past quarters.
- Forum rules: you are not allowed to post code that is part of your solution on the forum. An exception is a single line if it causes a compile-time or runtime error. Posting debugging output, including backtraces, is ok.
- You may not post detailed descriptions of your design or solution on the forum. You may not post answers to design document questions on the forum.
- Not having read the honor code and its stipulations is no excuse for violating it.
- If you have any doubt about what is and is not allowed, it is your obligation to ask the instructor beforehand.

Students with Disabilities

If you need adaptations or accommodations because of a disability (learning disability, attention deficit disorder, psychological, or physical), if you have emergency medical information to share with the instructor, or if you need special arrangements in case the building must be evacuated, please meet with the instructor as soon as possible.