CS 537 Notes, Section #1: Overview

Name, office, office hours, office phone, computer mailing address.

Mention lecture style with announcements in middle, then charge into material.

This course puts together things from many other courses: languages, architecture, hardware, data structures, algorithms.

"Operating system" is a hard term to define. What you consider an operating system depends on your needs and your view of the system.

WHAT IS AN OPERATING SYSTEM?

A scheduler/allocator:

- The operating system has resources for which it is in charge. Responsible for handing them out (and later recovering them).
- Resources include CPU, memory, I/O devices, and disk space.

A virtual machine:

- Operating system provides a "new" machine.

  This machine could be the same as the underlying machine. Allows many users to believe they have an entire piece of hardware to themselves.

  This could implement a different, perhaps more powerful, machine. Or just a different machine entirely. It may be useful to be able to completely simulate another machine with your current hardware. *Example of upgrading to a new piece of hardware. This can get out of hand. E.g., 1401 -> 360 -> 370 -> 3081.*

A multiplexor:

- Allows sharing of resources, and provides protection from interference.
- Provides for a level of cooperation between users.
- Economic reasons: we have to take turns.

According to these three views, if:

- we had enough hardware to give everyone too much;
- the hardware was well designed;

- the communications problem -- how to share knowledge -- is solved;

then we would not need operating systems. My view of operating systems says that they will still be needed:

As a servant and provider of services:

- Need to provide things like in the above views, but deal with environments that are less than perfect. Help the users use the computer by:

  providing commonly used subroutines;

  providing access to hardware facilities;

  providing higher-level "abstract" facilities;

  providing an environment which is easy, pleasant, and productive to use.

This view as a provider of services fits well with our modern network view of computing, where most resources are services.

What are the desirable qualities of an operating system? We can discuss them in terms of: Usability, Facilities, Cost, and Adaptability.

- Usability:
  - Robustness
    accept all valid input without error, and gracefully handle all invalid inputs
  - Consistency
    E.g., if "-" means options flags in one place, it means it in another. Key idea: ***conventions****. Concept: The Principle of Least Astonishment.*
  - Proportionality
    *Simple, cheap and frequent things are easy. Also, expensive and disastrous things (rm *) are hard.*
  - Forgiving
    Errors can be recovered from. Reasonable error messages. Example from "rm"; UNIX vs. TOPS.
  - Convenient
    Not necessary to repeat things, or do awkward procedures to accomplish things. Example copying a file took a batch job.
  - Powerful
    Has high level facilities.
- Facilities
  - Sufficient for intended use.
  - Complete.
    Dont leave out part of a facility. E.g., protection with
  - Appropriate.
    Do not use fixed-width field input from terminal.
- Costs

    - Want low cost and efficient services.
    - Good algorithms.
      Make use of space/time tradeoffs, special hardware.
    - Low overhead.
      Cost of doing nothing should be low. E.g., idle time at a terminal.
    - Low maintenance cost.
      System should not require constant attention.
- Adaptability
    - Tailored to the environment.
      Support necessary activities. Do not impose unnecessary restrictions. What are the things people do most -- make them easy.
    - Changeable over time.
      Adapt as needs and resources change. E.g., expanding memory and new devices, or new user population.
    - Extendible-Extensible
      Adding new facilities and features - which look like the old ones.

Why study operating systems? Two perspectives:

- Outside - depends on your level of sophistication.

  A system to compile and run Java programs

  Your average introductory Computer Sciences student.

  A system with many facilities - compilers, databases, file systems, system calls.

- Inside - internals, code, data structures.

  This is the system programmers view of an operating system. At this level you understand not only what is provided, but how it is provided.

Go over class information sheet. Initial programming assignment and chapter 1 from Dinosaur book.

Explain teaching and grading philosophy (probably when doing info sheet). Emphasize "come talk to me first" view before cheating.

QUESTIONS TO ASK ABOUT OPERATINGS SYSTEMS

Or, why are studying this stuff?

Why are operating systems important?

- They consume more resources than any other program.

  They may only use up a small percentage of the CPU time, but consider how many machines use the same program, all the time.

- They are the most complex programs.

  They perform more functions for more users than any other program.

- They are necessary for any use of the computer.

  When "the (operating) system" is down, the computer is down. Reliability and recovery from errors becomes critical.

- They are used by many users.

  More hours of user time is spent dealing with the operating system. Visible changes in the operating system cause many changes to the users.

Why are operating systems difficult to create, use, and maintain?

- Size - too big for one person

  Current systems have many millions lines of code. Involve 10-100 person years to build.

- Lifetime - the systems remain around longer than the programmers who wrote them.

  The code is written and rewritten. Original intent is forgotten (UNIX was designed to be cute, little system - now 2 volumes this thick). Bug curve should be decreasing; but actually periodic - draw.

- Complexity - the system must do difficult things.

  *Deal with ugly I/O devices, multiplexing-juggling act, handle errors (**hard!**).*

- Asynchronous - must do several things at once.

  Handles interrupts, and must change what it is doing thousands of times a second - and still get work done.

- General purpose - must do many different things.

  Run Skype, Java, Fortran, Lisp, FIFA Soccer, Databases, Web Servers, etc. Everybody wants their stuff to run well.

Operating systems are an unsolved problem.

- Most do not work very well.

  Crash too often, too slow, awkward to use, etc.

- Usually they do not do everything they were designed to do.
- Do not adapt to changes very well.

  New devices, processors, applications.

- There are no perfect models to emulate.

(No, UNIX/Linux is not it. Nor are Windows or MacOS. Maybe Plan 9?) Unlike fields like electronics where there are such models (zero distortion, flat response), any real system has (a large number of) flaws.
