CSE 451: Operating Systems
Winter 2021

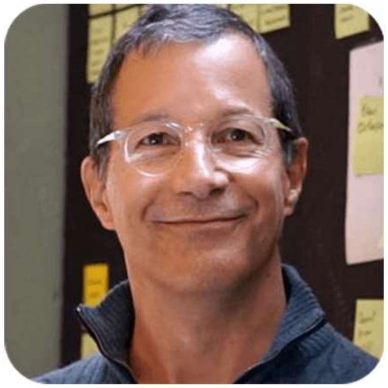
Module 1
Course Introduction

Gary Kimura
Mark Zbikowski

Course Staff

Instructors: Gary Kimura

Mark Zbikowski



TAs:

Tom Lou

Nicolas Monsees

Shivam Singhal

Brennan Stein

Yu Xin

First things first

This only a test…

- Occasionally test zoom’s robustness and Gary’s history of breaking things
- But what if we crash our zoom session?
- Chat Bomb: Where is everyone located?

Course Details

- Information portals
  - Course webpage, Canvas, Class discussion board
- Class Prerequisites:
  - CSE332: Data Structures and Parallelism
  - CSE333: Systems Programming
  - CSE351: The Hardware/Software Interface
- Lectures (will be recorded) – Gary and Mark
  - Use chat, raise hand, or unmute to ask a question
- Projects – Lead by the TAs
  - 4 labs using xk (brush up on programming in C). Posted all at once, due about every two weeks
- Grading (subject to change)
  - Projects 50%, Midterm 20%, Final 30%

The Projects

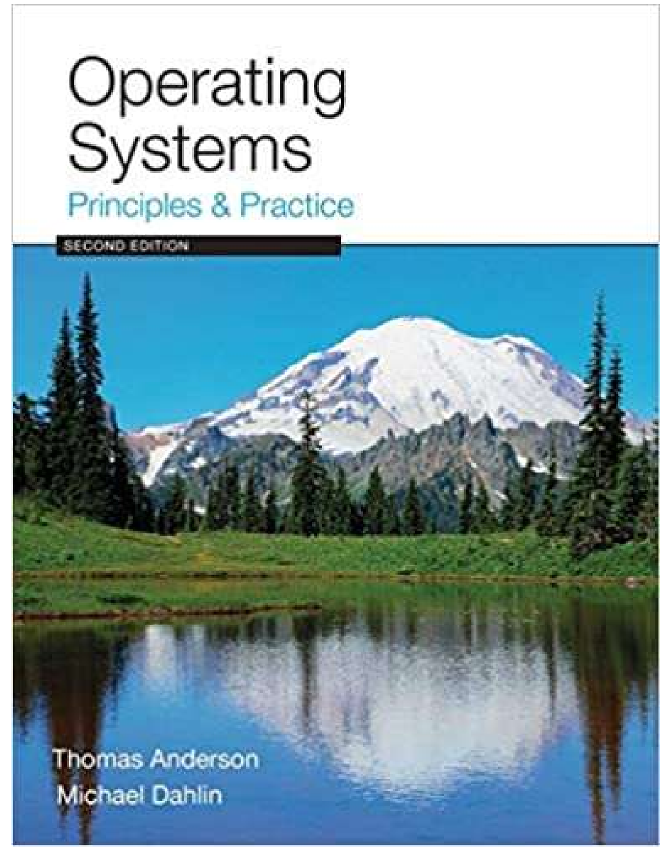
- Start them early
- Four of them
- Teams of two. You're likely to be happier if you form a team on your own than if we randomly assign them.
- Experience says that working remotely presents some challenges
  - Four projects instead of five
  - TAs are aware of this and are working on materials that could help
- Do not believe that passing the test cases means your code works...
  - Sorry
- Grading mechanism
  - TBD

Course Objectives

1. (One thing that I hope you learn from this class) How an OS is designed and built. To better know how to use the OS
2. Debugging large programs, adding new features to an existing (incomplete codebase)
3. Quote in the Cutler lab
   "Bugs: if you don't put them in, you don't have to take them out."
4. Drinking from the firehose for the first few weeks
5. Textbook: academic head knowledge
6. Lectures: enhance and supplement the textbook and do deep dives into some specific implementation issues.
7. Projects: get hand dirty, learn by doing

Course Roadmap (subject to change)

1. Chapter 2 Kernel Abstraction
2. Chapter 4 Concurrency
3. Chapter 5 and 6 Synchronization
4. Chapter 7 Scheduling
5. Chapter 8 and 9 Memory Management
6. Not in textbook Basic I/O organization
7. Chapter 11, 12, 13, 14 Storage
8. Wrap up loose ends



An important underlying concept
Policy vs. Mechanism

- Policy: is what you are trying to achieve
  - All the programs running on a computer get equal access to the CPU
- Mechanism: is how you achieve it
  - Use timers and context switching to share the CPU
- The lectures and textbook is mostly about policy, the projects deal with mechanism

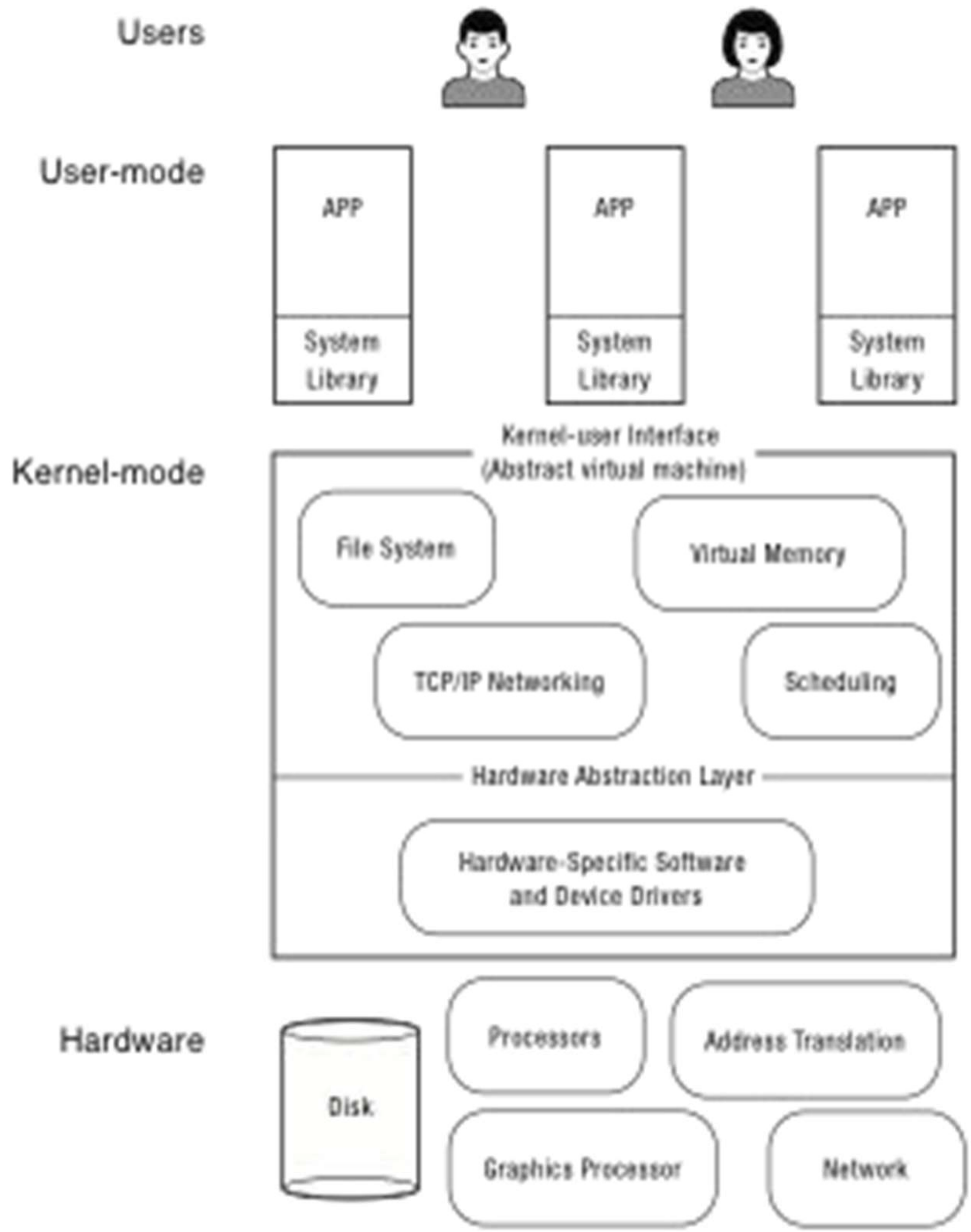
Operating System Mission Statement

- Class exercise, write an OS mission statement (e.g., The mission statement for the UW CSE School might be to prepare students in the art of computer usage for the coming century.)

One possible mission of the OS: provide a means for programs to effortlessly utilize the capability of the computer system

What is an operating system?

- Software to manage a computer's resources for its users and applications



Operating System Roles

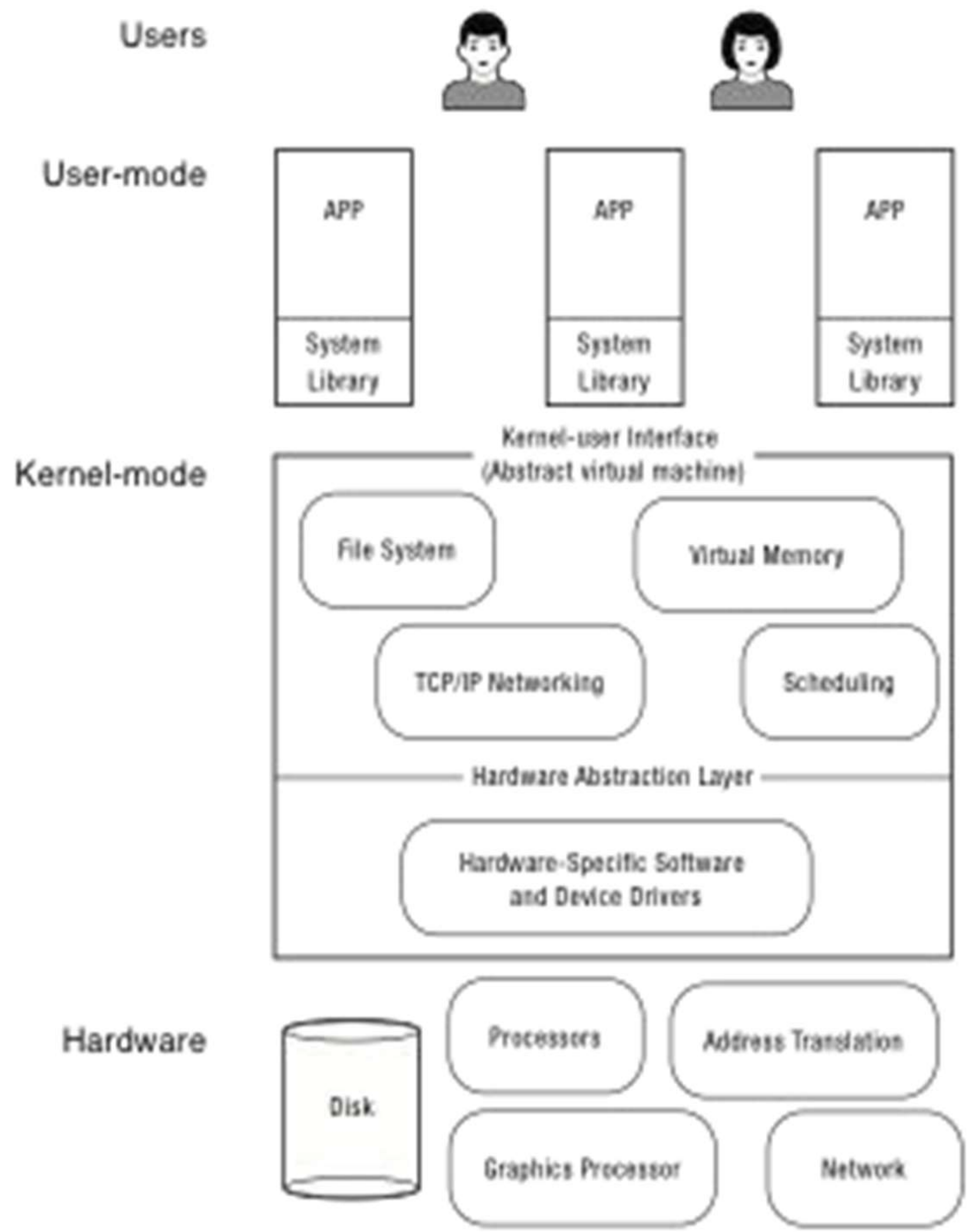
- Referee:
  - Ensures that everyone plays by the rules
  - Resource allocation among users, applications
  - Isolation of different users, applications from each other
  - Communication between users, applications
- Illusionist:
  - Each application appears to have the entire machine to itself
  - Infinite number of processors, (near) infinite amount of memory, reliable storage, reliable network transport
- Glue:
  - Libraries, user interface widgets, …

Example: File Systems

- Referee:
  - Prevent users from accessing each other's files without permission
  - Even after a file is deleting and its space re-used
- Illusionist:
  - Files can grow (nearly) arbitrarily large
  - Files persist even when the machine crashes in the middle of a save
- Glue:
  - Named directories, printf, …

OS Challenges

- Portability
  - For programs:
    - Application programming interface (API)
    - Abstract virtual machine (AVM)
  - For the operating system
    - Hardware abstraction layer



More OS Challenges

- Reliability
  - Does the system do what it was designed to do?
- Availability
  - What portion of the time is the system working?
  - Mean Time To Failure (MTTF), Mean Time to Repair
- Security
  - Can the system be compromised by an attacker?
- Privacy
  - Data is accessible only to authorized users

Even more OS Challenges

- Performance
  - Latency/response time
    - How long does an operation take to complete?
  - Throughput
    - How many operations can be done per unit of time?
  - Overhead
    - How much extra work is done by the OS?
  - Fairness
    - How equal is the performance received by different users?
  - Predictability
    - How consistent is the performance over time?
- Backward and Forward Compatibility
  - Can it run legacy apps?
  - How to accommodate growing or advancing Hardware. e.g., word size, or memory size.

Operating System History

- Batch to timeshare
- Single User to multiuser to single user (PC) to multiuser (servers, etc.)
- Cost of computer time compared to people time
- Single processor to multiple processors to distributed systems

Early Operating Systems: Computers Very Expensive

- One application at a time
  - Had complete control of hardware
  - OS was runtime library
  - Users would stand in line to use the computer
- Batch systems
  - Keep CPU busy by having a queue of jobs
  - OS would load next job while current one runs
  - Users would submit jobs, and wait, and wait, and

IBM 360 Mainframe

Time-Sharing Operating Systems: Computers and People Expensive

- Multiple users on computer at same time
  - Multiprogramming: run multiple programs at same time
  - Interactive performance: try to complete everyone’s tasks quickly
  - As computers became cheaper, more important to optimize for user time, not computer time

DEC PDP 11 (mini-computer)
(Tape Drive, Removable Hard Drive, Control Panel, Operator Console)

More PDP 11


digital equipment corporation · maynard, massachusetts
ADDRESS
DATA
pdp11/45

Today's Operating Systems: Computers Cheap

- Smartphones
- Embedded systems
- Internet of Things
- Laptops
- Tablets
- Virtual machines
- Data center servers

Tomorrow's Operating Systems: Everything gets Bigger*

- Giant-scale data centers
- Increasing numbers of processors per computer
- Increasing numbers of computers per user
- Very large scale storage

*And more compact

Computer Performance Over Time

	1981	1997	2014	Factor (2014/1981)
Uniprocessor speed (MIPS)	1	200	2500	2.5K
CPUs per computer	1	1	10+	10+
Processor MIPS/$	$100K	$25	$0.20	500K
DRAM Capacity (MiB)/$	0.002	2	1K	500K
Disk Capacity (GiB)/$	0.003	7	25K	10M
Home Internet	300 bps	256 Kbps	20 Mbps	100K
Machine room network	10 Mbps (shared)	100 Mbps (switched)	10 Gbps (switched)	1000
Ratio of users to computers	100:1	1:1	1:several	100+

- **But what hasn't increased over time?**
- **Have we reached certain limits?**

Next Up (Chapter 2 & 3) Kernel Abstractions

Roadmap for next few days

- Hardware modes
- Interrupts, exceptions, and syscalls (traps)
- Memory layout
- Booting the OS
- Processes and Process Management

The Dreaded Blue Screen