Introduction to CS350

Zille Huma Kamal

David R. Cheriton School of Computer Science
University of Waterloo

Spring 2022

Welcome!

Welcome to CS350 - Operating Systems!

- Administrative Information
- Introduction to Operating Systems

General Information

Important links:

- `http://www.student.cs.uwaterloo.ca/~cs350`
  Course personnel, office hours, readings, assignments, tutorials, previous midterms, review problems, etc.
- `https://piazza.com`
  Piazza will be used for announcements, extra notes, questions, corrections, etc. Please check piazza regularly. **Do not post your code in public piazza posts; use private posts when appropriate.**

Course Readings

Course notes are **required**.
They are **NOT** designed to be standalone. Come to class, take notes. Notes are available online from the course website. You may also purchase a printed copy, if you desire.

Textbook is **NOT** required, but highly recommended.
Operating Systems: Three Easy Pieces
Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
Textbook is available **FREE** on-line. Link to the text is available on course website. All recommended readings are linked on course website.

Grading Scheme

$A0$, $A1$, $A2$, $A3$: Assignment marks as a percentage
M: Midterm exam grade as a percentage
F: Final exam grade as a percentage

$\text{Normal} = 0.35 * A + 0.20 * M + 0.45 * F$
$\text{Exam} = (0.20 * M + 0.45 * F)/0.65$

```
if ( Exam < 50% ) {
    Course Grade = min (46, Normal, Exam)
} else {
    Course Grade = Normal
}
```

Note: you must pass the weighted average of the midterm and the final exam in order to pass the course.
Course outline on the course website includes an alternative grading scheme, if midterm and, or final exams have to be scheduled online.

Assignments

There are **4** assignments.
All assignments are to be done **individually**.

You will not be writing your own OS.
You will be adding/fixing features of an existing OS.

We use **OS/161** (~22,000 lines for kernel), which runs on **SYS/161** (MIPS simulator/VM)

Slip days:

- Allows flexibility in assignment deadlines
- Total of 5 slip days
- Can use maximum of 3 slip days per assignment

Continuity Plans in the face of Covid-19

- You should not come to class or attend any in-person activities if you are experiencing COVID-19 symptoms or are required to self-isolate. Here is playlist from Spring 2021 that you can use to follow along the lecture in case you miss class due to illness:
  - CS350 Spring 2021 Playlist from Dr. Lesley Istead
    Note, the material in Spring 2022 is slightly different from the Spring 2021 term. You are responsible for comparing the course notes from the course website for this term to ensure that you do not miss important material.
    To discuss any concepts that you may miss, feel free to come to virtual office hours
- In the event of absence due to influenza-like illness or required self-isolation, submit an Illness Self-declaration form in the Personal Information section in Quest.
- Contact the COVID-19 Support and Advice line to report their illness.

Plagiarism and Academic Offenses

READ AND UNDERSTAND INFO ON COURSE WEB PAGE
This course has extra requirements and ignorance is no excuse!

Do not use code from other sources:

- Do not copy code from friends, web sites, or other sources
- Do not search for or look at other code for any reason
- Avoid blogs that provide instructions
- We use VERY GOOD cheat detection software
- Every term people are caught
- Often: 0 on assignment and -5% off final grade

Plagiarism and Academic Offenses

Other than websites identified in the course, it is acceptable to use the web to

- understand the lecture material, learn how to use Git, bmake, GDB, and other tools used in this course

But it is not acceptable to use the web to

- get an idea of how to approach the assignment,
- copy or view code that may help you do the assignment

It is acceptable to consult with other students to

- get a **general** idea of how to approach the assignment
- get a **general** idea of how to overcome a stumbling block or fix a bug.

But it is not acceptable to

- view another student's code or have another student view your code.
- share more than general concepts/ideas
- write your discussion down

Plagiarism and Academic Offenses

IF you have taken this course before, you may reuse your previous code if:

- You ask your instructor for permission
- Your code was not subject to previous cheating penalties
- You understand it will be re-tested using our cheat detection software

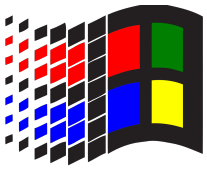
What happens when you ...

... "double-click" a program icon?
... save a file "foo.txt"?
... push a key on the keyboard?
... use `malloc`?
... print a file?
... use printf?

You will discover the answer to these and more this term!

What is an Operating System?

Generally, an OS is a system that:



- manages resources
- creates execution environments
- loads programs
- provides common services and utilities

Operating Systems

- originated 1951, 'LEO I' from J. Lyons and Co.
- started as simple I/O libraries, batch processors

Three views of an Operating System

Application View: what services does it provide?
System View: what problems does it solve?
Implementation View: how is it built?

An operating system is part cop, part facilitator.

Application View of an Operating System

The OS provides an execution environment for running programs.
The execution environment:

- provides a program with the **resources** that it needs to run, and
- provides **interfaces** through which a program can use networks, storage, I/O devices, and other system hardware components. Interfaces provide a simplified, abstract view of hardware to application programs.
- **isolates** running programs from one another and prevents undesirable interactions among them.

System View of an Operating System

The OS:

- **manages the hardware resources** of a computer system. Resources include processors, memory, disks and other storage devices, network interfaces, I/O devices such as keyboards, mice and monitors, etc.
- **allocates resources** among running programs.
- **controls the sharing of resources** among programs.

The OS itself also uses resources, which it must share with application programs.

Implementation View of an Operating System

The OS is a **concurrent, real-time** program.

- **Concurrency**, multiple programs/instructions running or appearing to run at the same time. Concurrency arises naturally in an OS when it supports concurrent applications.
- **Real-time**, programs that **must** respond to events within specific timing constraints. For example, hardware interactions impose timing constraints.

How does the OS implement these?

Definitions

kernel: The operating system kernel is the part of the operating system that responds to system calls, interrupts and exceptions.

operating system: The operating system as a whole includes the kernel, and may include other related programs that provide services for applications. This may include things like:

- utility programs
  - task managers
  - disk defragmenting tools
- command interpreters
  - cmd.exe
  - bash
- programming libraries
  - POSIX
  - OpenGL

Definitions

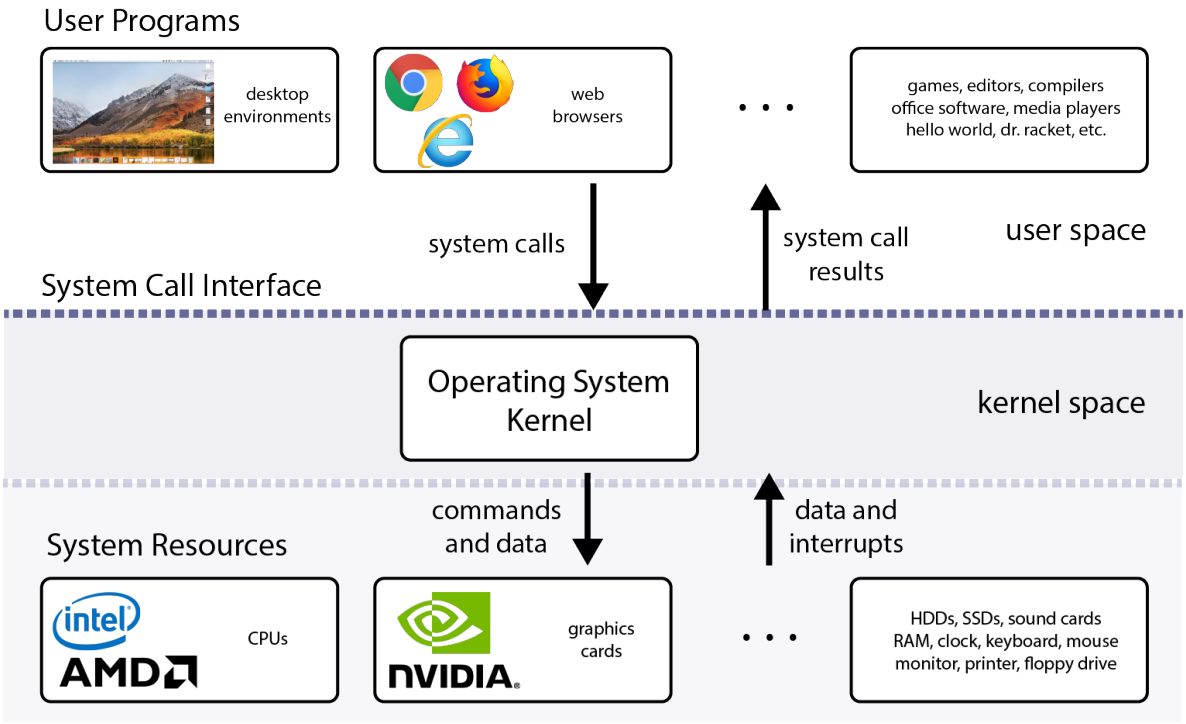
monolithic kernel: "everything and the kitchen sink" is a part of the kernel. This includes device drivers, file system, virtual memory, IPC, etc.

microkernel: only absolutely necessary components are a part of the kernel. All other elements are user programs.

real-time OS: an OS with stringent event response times, guarantees, and preemptive scheduling.

Windows, Linux, Mac OSX, Android and iOS are monolithic operating systems. They are **not** real-time. QNX is a real-time, microkernel operating system that originated here!

Schematic View of an Operating System



Operating System Abstractions

The **execution environment** provided by the OS includes a variety of **abstract entities** that can be manipulated by a running program. Examples of these abstractions:

files and file systems → secondary storage

address spaces → primary memory (RAM)

processes, threads → program execution

sockets, pipes → network or other message channels

This course will cover why and how these abstractions are:

- designed the way they are
- manipulated by application programs
- implemented by the OS

Course Coverage

- Introduction
- Threads and Concurrency
- Synchronization
- Processes and the Kernel
- Virtual Memory
- Scheduling
- Devices and Device Management
- File Systems
- Virtual Machines