Operating Systems - Introduction

Class	Operating Systems
Created	@Sep 2, 2020 12:22 PM
Materials	01 - Introduction.pdf
Reviewed	☐
Type	Lecture

- Professor' specialty is hardware/software working together for best performance
- What are the aspects of "best" performance?

  "Best" performance is not just fast, but also use of power, level of security, and reliability

Formal Goals of the Course

- These are the minimal goals we set out to achieve; the things that set us up for the best future
- What is an operating system?
  - Not anything GUI related, but the command line instruction asks instructions from an OS
- How does the OS interact with hardware and other software applications?
  - No matter what OS you use, it is the only thing that can use the hardware
  - Software "asks" OS for hardware on its behalf and the OS does the heavy lifting

- It is key to design an OS that doesn't require change post-release, such as a 3D printer needing to connect to it for a given reason
- Memory management is all abotu delegation for current tasks
- File systems are about opening and closing files, not just reading and writing (which is easy when it's a few lines of code)

Informal Goals of the Course

- Go beyond the A to truly understand the concepts taught
- As long as you show up and show understanding of the concepts, putting forth the proper effort, you will achieve that A
    - It's based on a curve of sorts: if you get a 51% and everyone else gets 50% you'll be fine, but if you get a 95% and everyone gets a 99% well... not so much
- Learn OS and enjoy doing so, the prof is all about making it an interesting topic and fun to learn
- Don't overthink the exams, there aren't any tricks or complicated concepts
    - Exams are one of many ways to display understanding and *that is what they are meant to do*
    - There are 3 (maybe 4) labs which is nice
- To be able to use what you have learned in a variety of different concepts, connecting more dots and not becoming specialized in one area
- To acquire the knowledge to develop an OS, or parts of it, should you choose to do that
- To start research projects related to OS
    - Really learning and understanding, not just rote studying
- We do not use Java or Python, it is always C or C++
    - This is a perfect opportunity to launch C/C++ knowledge forward in a learning environment and get *very uncomfortable* with it!

    ▼ Why do we not use Python? What is an interpreted language?

Python is an interpreted language, and this is very slow. An interpreted language compiles the code line by line instead of compiling the entire thing and then running.

▼ What are the two main reasons we do not use Java?

We do not use Java because (1) it compiles code into bytecode, which requires a JVM to process it and run it, and (2) because Java manages memory (such as garbage collection) which takes control away from the programmer.

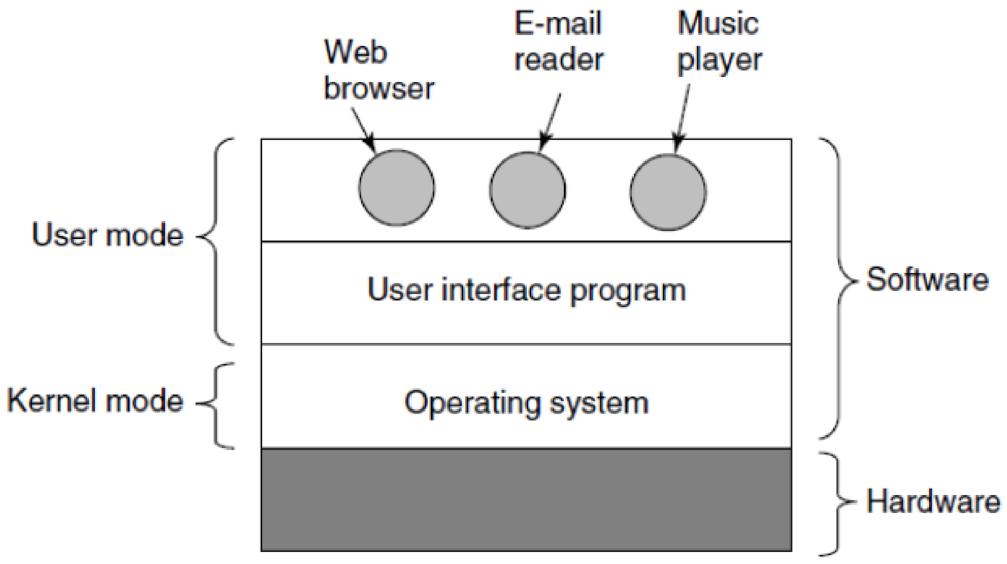
Web Presence

- Study using the forums ofr questions that others have asked- you never know if there's something you didn't think about
- All assignments are submitted via NYU classes
- Slides posted about one or two days ahead of lectures
  - Don't have to review these ahead of time, as I'm more concerned with being caught up
- Readings for a lecture are due on the day labeled on the syllabus, not assigned that day
  - Readings will always be review, the *slides are the primary source for material*
- There is no focus on memorization, just learning how to properly think through problems
  - Focus on learning and understanding, not studying and memorizing

The Interesting Things

▼ What two "levels" are bridged by the operating system?

At the lowest level you have your hardware (GPUs, RAM, etc.) and at the highest level you have the software. The OS bridges these two levels.



- As a programmer you deal with user mode, and you call functions that then deal with the OS which then communicates with the hardware
- The compute ris what is responsible for switching between these modes
- ▼ What does the kernel (mode) refer to?

  The kernel is the code of the OS

Two Main OS Tasks

1. Provide programmers (and programs) a clean abstract set of resources
2. Manage hardware resources

- ▼ What does the term abstract refer to?

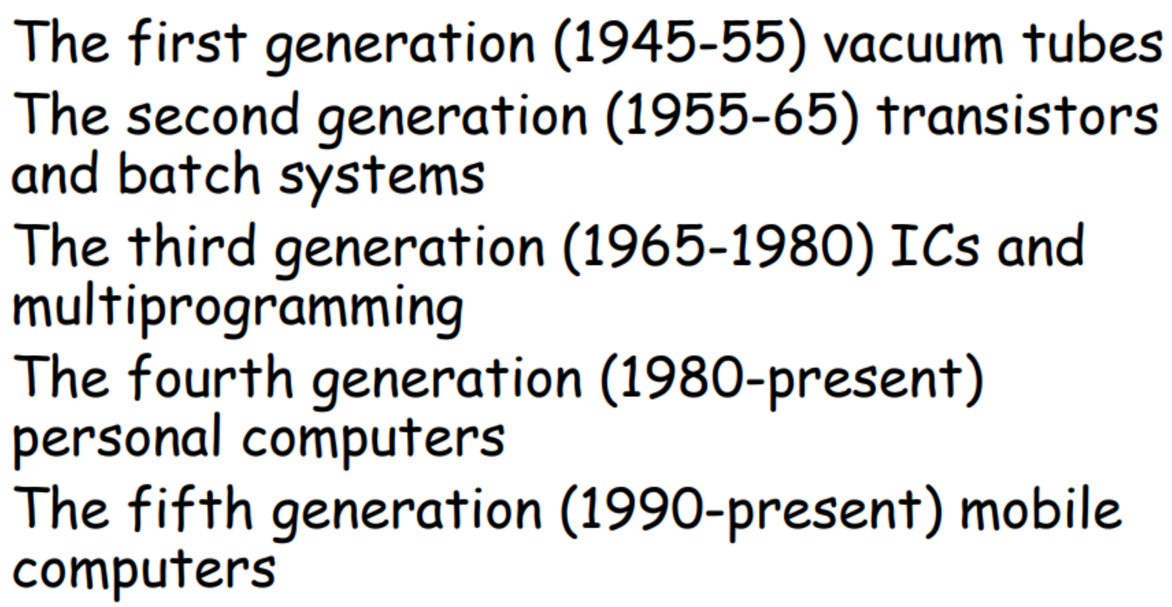
  I think it refers to a set of resources that are not directly related to each other but also have more meaning then random memory addresses and components

- In general, multiple programs running at once
  - ▼ What is the relation between programs and processes?

    Once a program is run/executed, it becomes a process
  - When there are multiple processes running, the OS needs to manage hardware resources. The dealing with resource management is largely the kernel's job
- Dynamic memory allocation varies, calling the OS is a slow process. More on this later in the course
- Applications to beautiful interface to OS to ugly interface to hardware
  - Writing a lin ecommand versus finding the proper spot on the drive to write data to a specific location on the disk

History of OS

▼ Specific dates from slides



The first generation (1945-55) vacuum tubes

The second generation (1955-65) transistors and batch systems

The third generation (1965-1980) ICs and multiprogramming

The fourth generation (1980-present) personal computers

The fifth generation (1990-present) mobile computers

▼ What are transistors and what preceeded them?

Transistors are the basic building blocks of computers, and are physically just switches. They were preceeded by vacuum tubes (that had a constantly running life of about 18 hours)

- OS used to be a human job, people would run around dealing directly with the hardware components
- Intel arose in 1969, IBM stands for International BUsiness Machine, Intel stands for Integrated Electronics, AMD is another one…

Three Easy Pieces

1. Virtualization

   Allowing programs to run as if they have all the memory and storage of the entire computer, even though they do not. Imagine having to write a program and dealing with the possibility that other programs are using up resources.

2. Concurrency

   Running multiple things at once is an important job and largely relates to parallel programs, multithreading, etc.

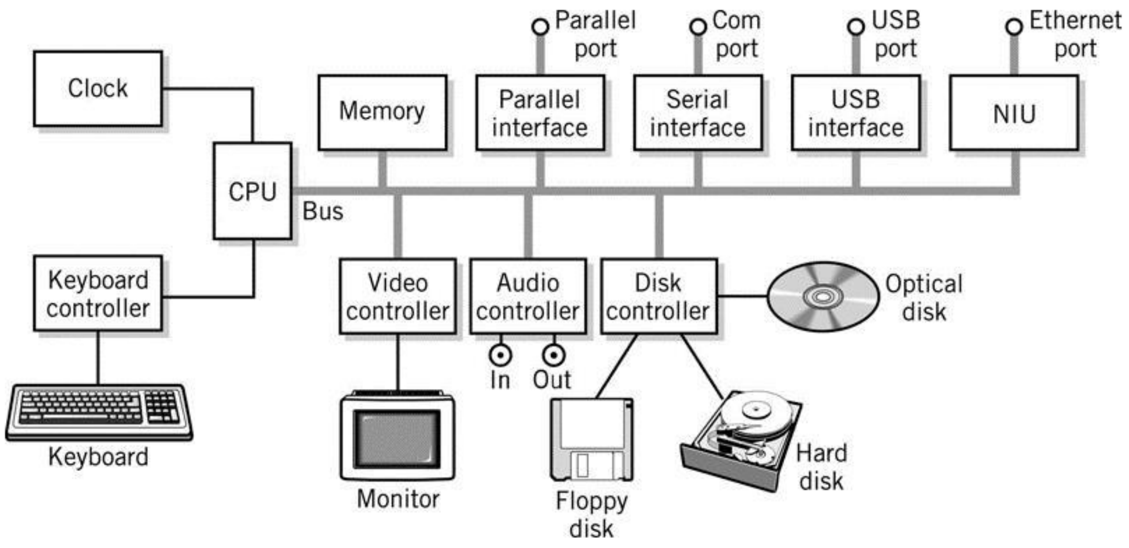
3. Persistence

   The ability for things to persist beyond restarts or crashes, such as files and data

---

- The three easy pieces assist the previous two major tasks of an OS

▼ Typical Computer System diagram



Booting Sequence

1. The **BIOS** starts up
    - Has its own battery, checks RAM, keyboard, and other devicies- a process called POST
2. BIOS then determines the **boot device**
3. The first **sector** in boot device is read into memory and excuted to determine the **active partition**
4. The **secondary boot loader** from that partition is then used to load the OS from the active partition

▼ What does POST stand for?

Power On Self Test

▼ What is the active partition?

It is the component of a drive that contains the OS that the boot loader is meant to find

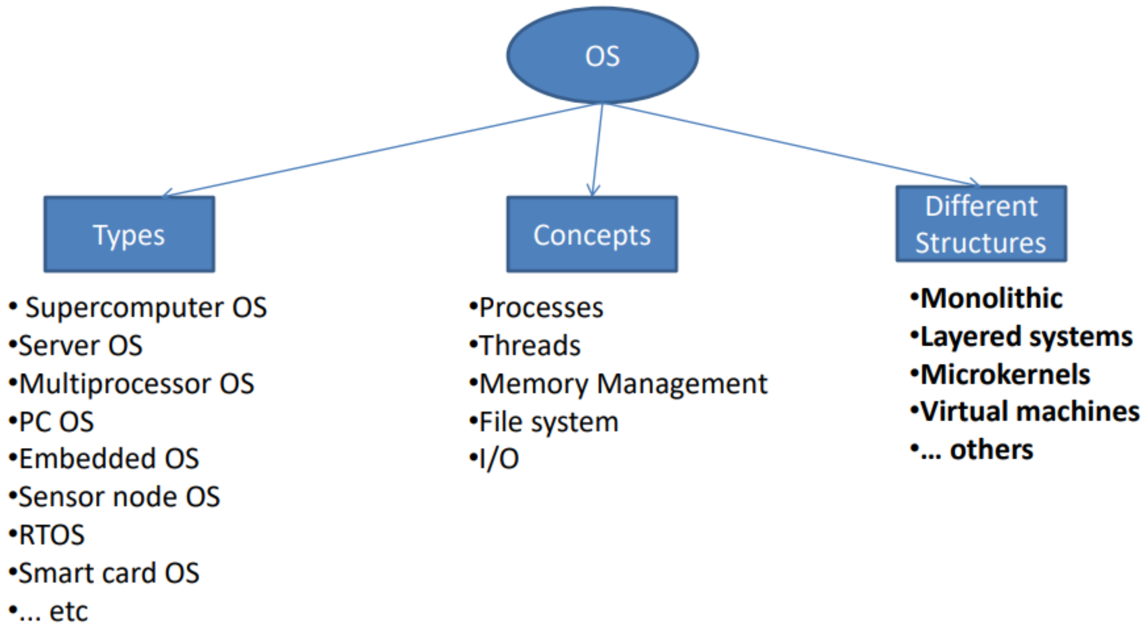
▼ Why is the BIOS not considered an OS?

It doesn't deal directly with the user (usually) and is just a mini device that finds the OS

- When errors occur, the BIOS just beeps (i.e. no screen found)

General OS "Parts" for Learning

▼ Types, Concepts, Different Structures — Can you label a few of each?



▼ What are the main objectives of an operating system?

1. Convenience
2. Efficiency (related to "best")
3. Ability to evolve (such as with patches)

- USER ↔ Processes ↔ OS ↔ Hardware

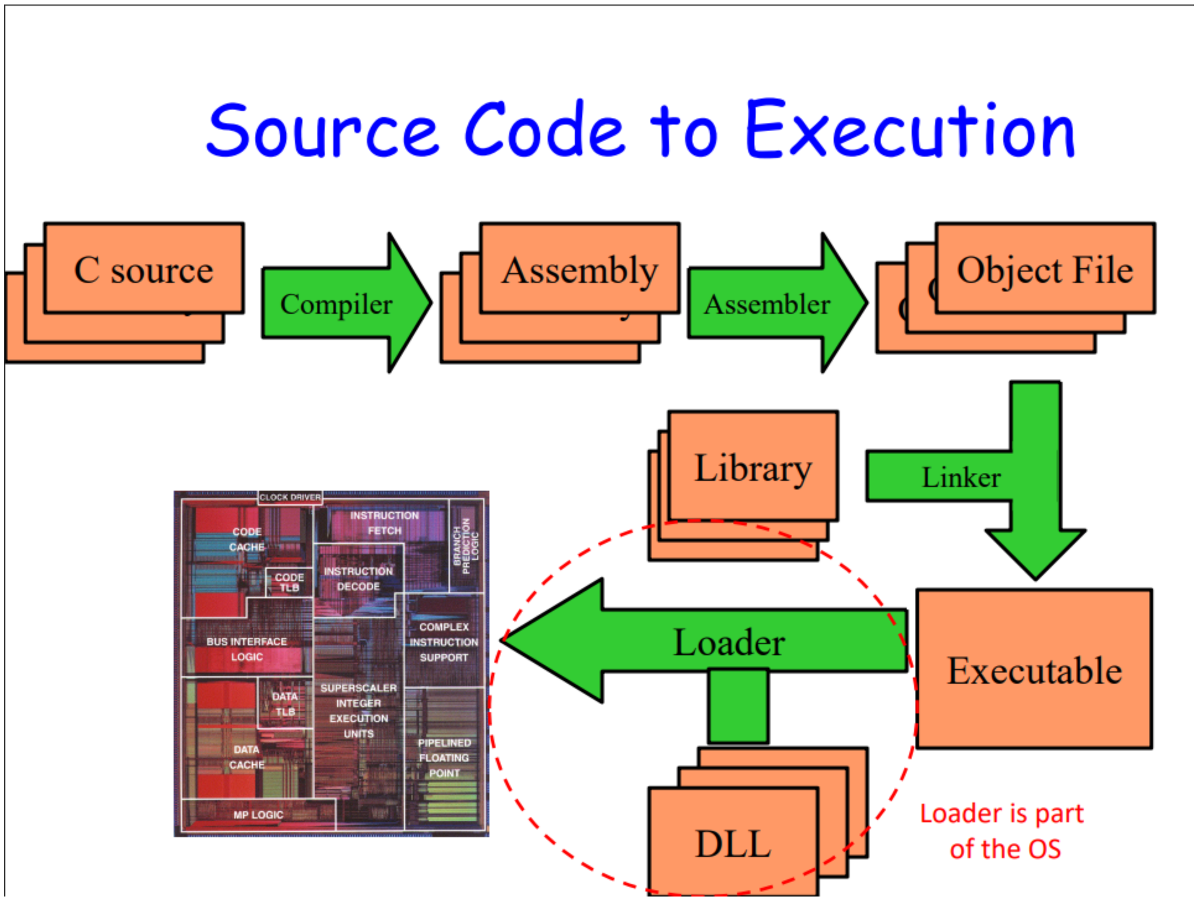
A Process

- **Definition:** A program in execution
- Key concept in all operating systems
- A process is associated with an address space
  - ▼ What is an address space?

    An address space is something which defines a range of [memory] addresses
- A process is also associated with a set of *resources* and can be seen as *containers* of sorts
  - ▼ What does container mean in this context?

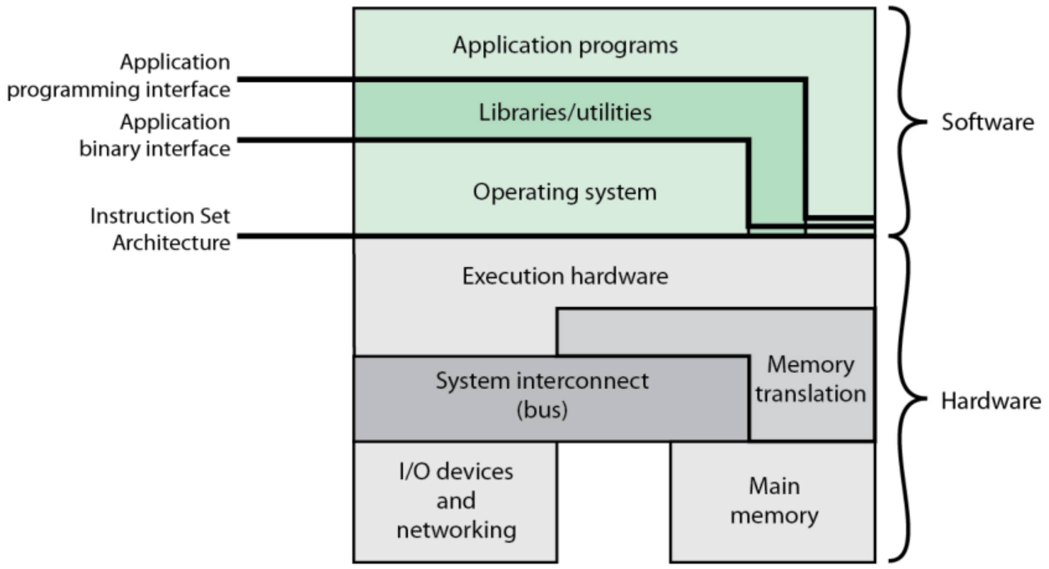
Something that holds all the needed information. In this case the process holds all the necessary information it needs to run itself.

▼ Source code to execution

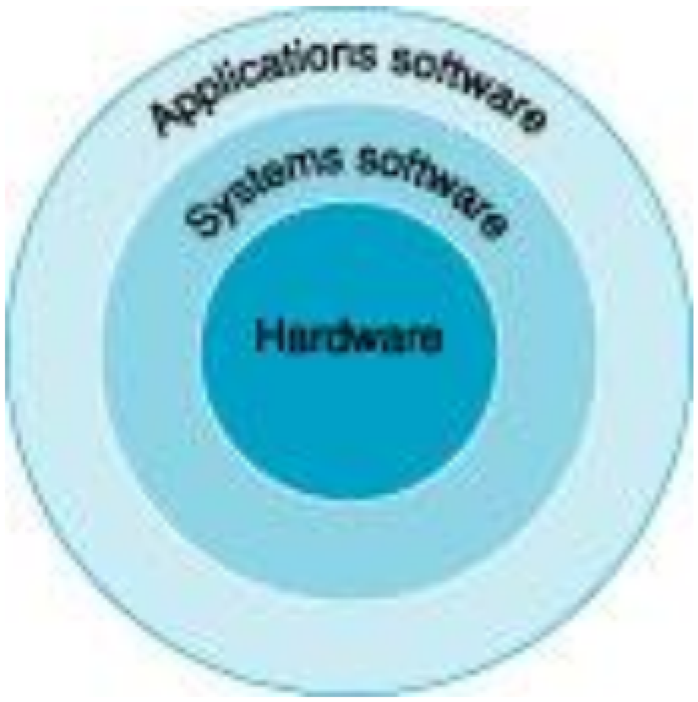


▼ Hardware and Software Infrastructure

Hardware and Software Infrastructure


Application programs
Application programming interface
Libraries/utilities
Application binary interface
Operating system
Instruction Set Architecture
Software
Execution hardware
Memory translation
System interconnect (bus)
Hardware
I/O devices and networking
Main memory

Computer Hardware and Software Infrastructure



Wrap-Up

- At its core, OS is a manager
- OS works in different environments under a variety of restrictions