Introduction and History

Sarah Diesburg

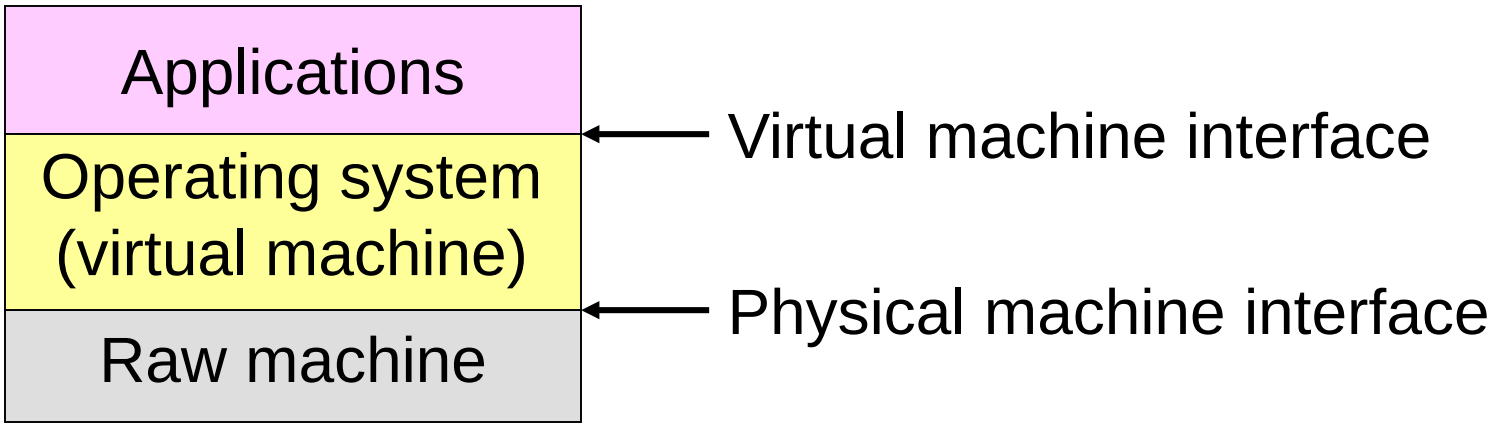
Operating Systems

CS 3430

What is an Operating System?

What is an Operating System?

- A virtual machine
  - Hides the complexity and limitations of hardware from application programmers



For Each OS Component

- There are two major design questions:
  - What is the hardware interface?
    - The physical reality
  - What is the application interface?
    - The nicer abstraction

Reality vs. Abstraction

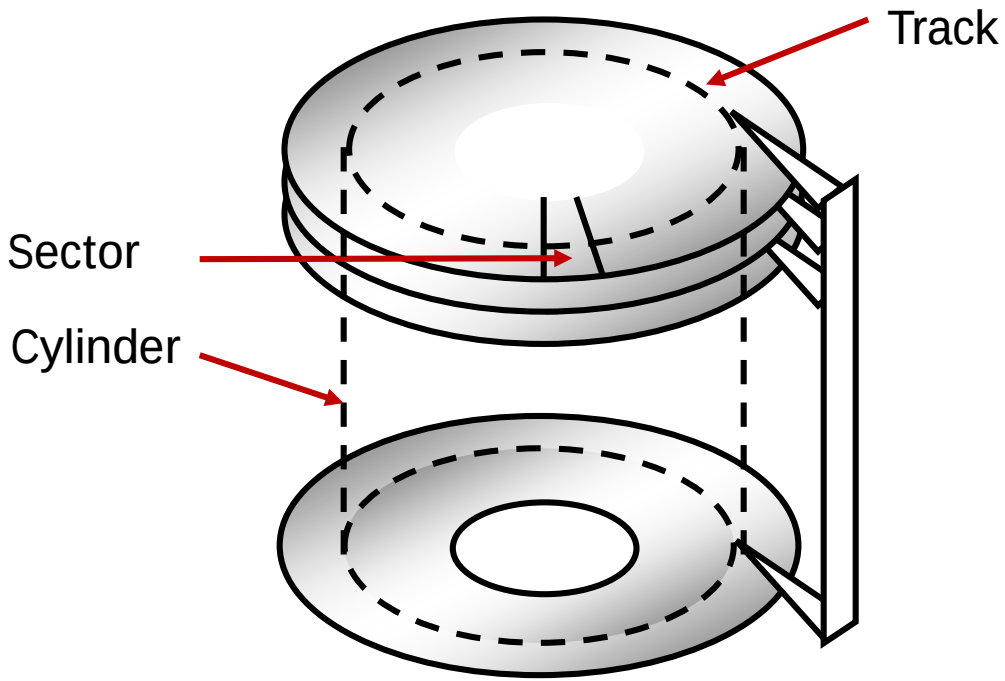
Reality	***Abstraction***
Multiple CPUs	A single CPU
Limited RAM capacity	Infinite capacity
Mechanical disk	Memory speed access
Insecure and unreliable networks	Reliable and secure
Many physical machines	A single machine

Two General OS Functions

- ***Standard services***
  - Screen display
  - Disk accesses
- ***Coordination among applications***
  - Goals: correctness, efficiency, and fairness

Standard Services

- Example: disk drive



Disk Access

- Raw disk access
  - `write(block, len, device, track, sector);`
- OS-level access
  - `lseek(file, file_size, SEEK_SET);`
  - `write(file, text, len);`

Coordination

- Example: Protection
  - Applications should not crash one another
    - ***Address space***: all memory addresses that an application can touch
  - Applications should not crash the OS
    - ***Dual-mode operations***
      - ***Kernel mode***: anything goes
      - ***User mode***: an application can only access its own address space

Four Recurring Themes

- OS as an illusionist
  - Overcomes hardware limitations
- OS as a government
  - Protects users from one another
  - Allocates resources efficiently and fairly
- OS as a complex system
- OS as a history teacher
  - Learns from the past to predict the future

History of OS: Change!

		1980	Current	Factor
Speed	CPU	1 MIPS	146,000 MIPS	1.5×10^5
	Memory	500 ns	0.9 ns	5.6×10^2
	Storage	18 ms	30 μs	6.0×10^3
	Network	300 bits/sec	100 Gb/sec	3.6×10^8
Capacity	Memory	64 Kbytes	128 GB	2.0×10^6
	Disk	1 Mbytes	4 TB	4.0×10^6
Cost	Per MIP	$100K/MIP	$.0076/MIP	1.3×10^7
Other	Address bits	8	64	8
	Users/machine	10s	0.01	1.0×10^3

History Phase I: Hardware Expensive, Humans Cheap

- Hardware: mainframes
- OS: human operators
  - Handle one ***job*** (a unit of processing) at a time
  - Computer time wasted while operators walk around the machine room

OS Design Goal

- Efficient use of the hardware
  - ***Batch system***: collects a batch of jobs before processing them and printing out results
    - Job collection, job processing, and printing out results can occur concurrently
  - ***Multiprogramming***: multiple programs can run concurrently
    - Example: I/O-bound jobs and CPU-bound jobs

History Phase II: Hardware Cheap, Humans Expensive

- Hardware: terminals
- OS design goal: more efficient use of human resources
  - ***Timesharing systems***: each user can afford to own terminals to interact with machines

History Phase III: Hardware Very Cheap, Humans Very Expensive

- Hardware: personal computers
- OS design goal: allowing a user to perform many tasks at the same time
  - ***Multitasking***: the ability to run multiple programs on the same machine at the same time
  - ***Multiprocessing***: the ability to use multiple processors on the same machine

History Phase IV: Distributed Systems

- Hardware: computers with networks
- OS design goal: ease of resource sharing among machines
  - Example: Chrome OS is almost a pure, locked-down "web thin client", where almost all apps require the Internet

The Bottom Line

OS designs need to adapt to changing technology