

## CS414 Operating Systems Prof. Marty Humphrey

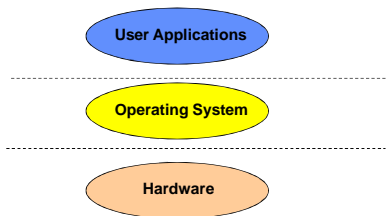
CS414: Operating Systems

## Preliminaries

- **My apologies about canceling Tuesday's class**
  - Class decided: Make-ups: Thurs 6:00 – 7:15 \*AND/OR\* Frid 3-4:15
  - Currently I'm scheduled to miss 4 class meetings
- **Peer notetaker**
- **Intro class survey at the end today (10 min)**
- **Group questions of the day (5 min)**
  1. What is the purpose of an operating system?
    - Variation: what does the OS do *for you*?
  2. When has an operating system *failed* you ("done something wrong")? Examples?
  3. Why is this a required class?
    - Variation: How many of you think you're going to someday provide code for an OS such as Linux and/or Windows?
  4. You're a SW developer for a customer: why might you choose one OS vs. another?

CS414: Operating Systems

## What is an Operating System?



Hmmm... that's it? This doesn't seem so tough. Why are we here?

CS414: Operating Systems

## So... Q1. What does an Operating System provide?

- **Persistent storage (programs, data) in file system**
- **"Compartmentalization" of running programs**
  - Scheduling
  - IPC
  - Protection/Isolation
  - *Sharing*
- **Illusion of infinite memory**
  - "new" in C++ will *probably not* fail
- **Uniform API for programmers**
- **Low-level ability to talk with other machines**
- **(From ordinary user perspective) Stable, predictable environment for running apps**
- **Others....**

CS414: Operating Systems

## Another take...

- **Provides Abstractions**
  - Avoid complicated, idiosyncratic interfaces of low-level physical devices by using *abstractions*, which have clean interfaces; Examples: processes, unbounded memory, files, synchronization and communication mechanisms
- **Provides Standard Interfaces**
  - Goal: portability
- **Mediates Resource Usage**
  - Share resources fairly, efficiently, safely, securely
- **Consumes Resources**
  - It takes money to make money
- **Complicated**
  - An OS is not a trivial piece of software (must support concurrency and asynchrony)
- **Purpose**
  - Can be general purpose (e.g. timesharing) or special purpose (e.g. process control and real-time or embedded systems—must be small and predictable)
  - Domains: floodgate control, automobile brakes, space shuttle; "time at which computation occurs can be as important as logical correctness")

CS414: Operating Systems

## Summary: Multiple Views of OS

- **OS as juggler**
  - provide illusion of "more machine" than actually exists (*virtual machine*)
- **OS as government**
  - protect users from each other, allocate resources efficiently and fairly, provide safe and secure communication
- **OS as complex system**
  - keeping OS design and implementation as simple as possible

CS414: Operating Systems

## Q2. When has an OS failed you?

- Blue Screen of Death (BSOD)
- Not given me enough of the virtual machine (e.g., multi-user)
- Lost my data (either in-core or in files)

CS414: Operating Systems

## Q3. Why/What?

1. Why are we still studying this – aren't OSs a "solved problem"?
2. What makes one OS (e.g., linux) different than another OS (e.g., Windows)?
  1. What makes an OS for a cell phone different than an OS for a supercomputer?
3. When I write a program, how much do I "use the programming language" and how much do I "use the OS" and how much of it is "hardware"?
4. Is the OS always running?
5. If the CPU can execute 1+ billion (1E9) instructions/sec, then why does "hello world" take so long? (e.g., 1 millisecond via "time", 80 microseconds via `rdtsc`)
  1. "hello world" requires 1 million instructions? `Printf` requires 116979 clock ticks?

CS414: Operating Systems

## Why/What? (cont.)

6. Windows XP is 45 million lines of code, RH Linux 7.1 has 30 million – WHY?
7. Why does my OS sometimes get "wedged"?
8. Why are there more than one file system types?
9. What is the hourglass on Windows?
  1. What is it WAITING for?
10. What are the differences between Windows and Linux?
  1. Why does Linux exist?

CS414: Operating Systems

## Are modern-day OSs "designed properly"?

UNIX (1970s),  
Linux is based on  
UNIX; Windows NT  
(1980 or so)      Vista (but is it built  
from scratch)?

	1983 (desktop)	2007 (e.g., my laptop)
MIPS	0.5	1600 x 2
Price/MIP	\$100,000	\$.21
memory	1 MByte	1.25 Gbyte
Network	10 Mbps	100 Mbps – 1 Gbps
Backing store (tape)	1 GByte	1 TeraByte (120G Disk)
Addressable bits	32	32 or 64

What other area has seen such orders of magnitude change? (Answer: none)

CS414: Operating Systems

## Logistics/Syllabus

- <http://www.cs.virginia.edu/~humphrey/cs414>
- Programming assignments
  - Basic setup is based on virtual machines
  - Some Linux, some Windows Research Kernel (WRK)

CS414: Operating Systems

## Approach of class

- Slides will NOT EXACTLY follow text
  - Why? Provides a different perspective on material
  - Therefore: you must read text as well as course materials
- Slides will be posted AFTER class
  - Why? Slides are prepared Just-in-time (JIT)
- What you will learn (balance three things)
  1. What an OS does ("theory")
  2. How an OS is implemented ("practical experience")
    1. Windows Research Kernel and Linux mods
    2. Poking at Windows/Linux from "user-level"
  3. How the OS impacts your user programs ("exploit the OS for your own purposes")

CS414: Operating Systems

## Next time

- **Purchase the book**

- OS by Silberschatz – [thecheapestbook.com](http://thecheapestbook.com) or something similar

- **Look at the class web page**

- **Read Chap 1 and Chap 2 for Tuesday**