



Operating Systems

Introduction

Topics

- ♦ What is an OS?
- ♦ OS History
- ♦ OS Concepts
- ♦ OS Structures

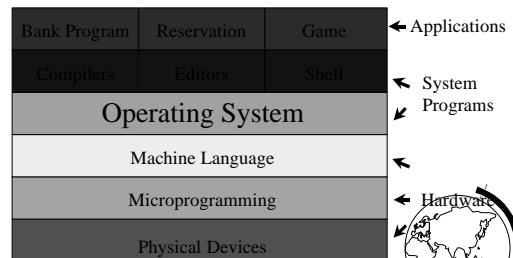


Let's Get Started!

- ♦ What are some OSes you know?
- ♦ Pick an OS you know:
 - What are some things you like about it?
 - What are some things you don't like about it?



What is an Operating System?



What is an Operating System?

- ♦ An Extended Machine (Top-down)
 - Transforming - new resource
 - ♦ ex: Win98 device manager
- ♦ A Resource Manager (Bottom-up)
 - Multiplexing - illusion of several resources
 - ♦ ex: browse the web AND read email
 - Scheduling - deciding who gets what when
 - ♦ ex: compile fast OR edit fast
- ♦ Why have an OS?
 - Convenient and Efficient
 - ♦ Programming hardware difficult
 - ♦ Idle hardware "wasteful"



OS History

- ♦ Helps understand key requirements
 - Not one brilliant design
 - ♦ (despite what Gates or Torvalds might say)
 - Fixed previous problems, added new ones
 - Tradeoffs
- ♦ Closely tied to:
 - Hardware history
 - User history



Hardware History

	1981	1999	Factor
Power (SPEC)	1	250	250
\$/Power	\$100K	\$45	2200
Memory	128K	128M	1000
Disk Capacity	10M	10G	1000
Net Bandwidth	9600b/s	155Mb/s	15000
Users / Mach.	10s	<=1	10

♦ Comments? Change!



Hardware Very Expensive Humans Cheap

- ♦ Single program execution (no OS)
- ♦ Hardwire “programming”
- ♦ Programming slow, not “offline”!
 - Punch cards



Hardware Very Expensive Humans Cheap

- ♦ Punch cards
- ♦ Fortran or assembler
- ♦ Waste computer time walking!
 - Batch programs on tape



Hardware Very Expensive Humans Cheap

- ♦ Programs read in from tape
- ♦ Two applications:
 - Scientific
 - Data processing
- ♦ CPU idle during I/O!
 - Multiprogramming with partitions
 - Spooling as jobs finished



Hardware is Cheap Humans Expensive

- ♦ Turn around time 1/2 day
- ♦ Programmer time wasted!
 - “Sigh. In the good old days....”
 - Time-sharing
 - Multics (sorta)
 - New problems
 - ♦ response time
 - ♦ thrashing
 - ♦ file-systems



Hardware Very Cheap Humans Very Expensive

- ♦ Personal computers
 - Network operating systems
 - Distributed operating systems
- ♦ OSes today
 - small == 1000K (15 pages, 5 programmer years)
 - large == 10,000K (150 pages, 500 programmer years) (longer than a semester :-))
 - need to evolve quickly
 - ♦ hardware upgrades, new user services, bug fixes
 - efficient and/or modular kernels



Windows NT History

- ♦ 1988, v1
 - split from joint work with IBM OS/2
 - Win32 API
- ♦ 1990, v3.1
 - Server and Workstation versions
- ♦ 1997(?), v4
 - Win95 interface
 - Graphics to kernel
 - More NT licenses sold than all Unix combined



Windows NT Today

- ♦ Microsoft has 80% to 90% of OS market
 - mostly PC's
- ♦ 500+ MHz Intel Pentium
- ♦ NT aiming at robust, server market
 - network, web and database
- ♦ Platforms
 - Intel 386+ - Alpha
 - MIPS R4000 - PowerPC
 - (Dropping 64-bit)
- ♦ (Win2000 merges Win98 and WinNT)



Linux History

- ♦ Open Source
 - Release Early, Release Often, Delegate
 - “The Cathedral or the Bazaar”
- ♦ Bday 1991, Linus Torvalds, 80386 processor
 - v.01, limited devices, no networking,
 - with proper Unix process support!
- ♦ 1994, v1.0
 - networking (Internet)
 - enhanced file system (over Minix)
 - many devices, dynamic kernel modules



Linux History

- ♦ Development convention
 - Odd numbered minor versions “development”
 - Even numbered minor versions “stable”
- ♦ 1995, v1.2
 - more hardware
 - 8086 mode (DOS emulation) included
 - Sparc, Alpha, Mips support started
- ♦ 1996, v2.0
 - multiple architectures, multiple processors
 - threads, memory management



Linux Today

- ♦ v2.2
- ♦ 1,000,000 lines of code
- ♦ 7-10 million users
- ♦ Estimated growth 25%/year through 2003
 - all others, 10% combined



Where are we?

- ♦ Ch 1-3 by Monday
 - Reading details on course Web page
 - Ch 1, brief, alternate viewpoint
 - Ch 2, computer architecture review
 - Ch 3, today and Monday
- ♦ Timeline on Web page
 - Proj 0 in by Monday
 - Proj 1 out Monday
 - HW 1 out Tuesday



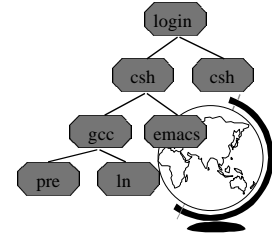
Operating System Concepts

- ♦ Processes
- ♦ Files
- ♦ System Calls
- ♦ Shells



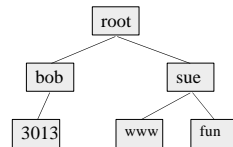
The Process

- ♦ Program in execution
- ♦ Running -> Suspended -> Running
- ♦ Example: the Shell
- ♦ Process “Tree”
- ♦ Signals
- ♦ UID (GID)
- ♦ (Two weeks)



Files

- ♦ Store data on disk
- ♦ Directory “Tree”
- ♦ Working directory
- ♦ Protection bits
 - 9 in Unix: **rwX** bits, ex: `rwXr-x--x`
- ♦ Abstraction of I/O device
 - terminal, printer, network, modem
- ♦ Pipe
- ♦ (1-2 Weeks)



System Calls

- ♦ Way processes communicate with OS
- ♦ example:


```
write(file, string, size)
```
- ♦ OS specific!
- ♦ POSIX (1980s)
 - Portable Operating System (unIX-ish)
- ♦ (Most of the projects)



Shells

- ♦ User’s interface to OS
- ♦ Simple commands
 - “cd”, “cat”, “top”
- ♦ Modifiers
 - ‘&’, ‘|’, ‘>’
- ♦ (Project 1 is to write a shell)



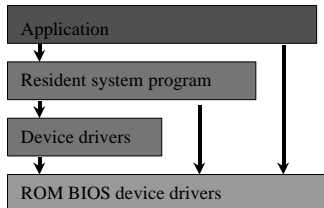
Operating System Structure

- ♦ Simple Systems
- ♦ Virtual Machines
- ♦ Micro Kernels



Simple Systems

- ♦ Started small and grew, no hardware support
- ♦ MS-DOS

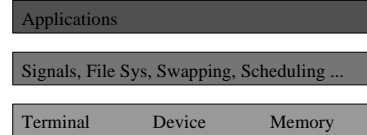


♦ Protection!



Simple Systems

- ♦ Unix (see /vmunix)



- ♦ “The Big Mess”
- ♦ Some move towards a more modular kernel



Virtual Machines

- ♦ IBM VM/370

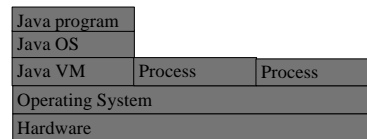
Process	Process	Process
Process	Process	Process
Operating Sys	Operating Sys	Operating Sys
Virtual Machine		
Hardware		

- ♦ Complete protection
- ♦ OS development, emulation
- ♦ Performance!



Virtual Machines

- ♦ Java Virtual Machine

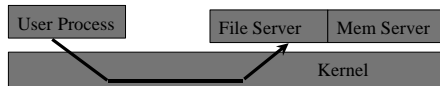


- ♦ Platform independence!



Micro Kernel

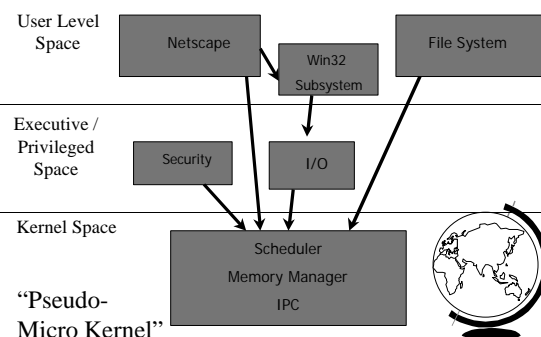
- ♦ Mach



- ♦ Client-Server
- ♦ Good performance
- ♦ Adaptable to distributed OS
- ♦ Robust
- ♦ Careful about mechanism!

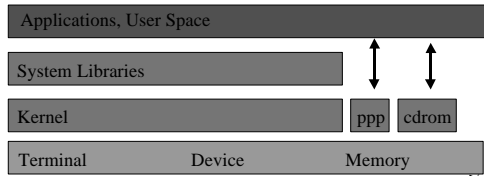


WinNT Structure



Linux Structure

◆ “Simple” system



◆ Loadable Modules

- done after “boot”
- allow 3rd party vendors
- easier for development

