Today’s topics

Virtual Environment for Computing
Operating Systems

Upcoming
Program Execution Times
(Great Ideas, Chapter 13)

Reading
Great Ideas, Chapter 10

The Problem

- The Raw Machine Provides a Hostile Environment
  - Imagine program in machine language (on real machine).

Machine Language Program for MIPS Machine

- [00400000] 8fa40000 lw $4, 0($29) ; lw $a0, 0($sp)
- [00400004] 27a50004 addiu $5, $29, 4 ; addiu $a1, $sp, 4 #
  argv
- [00400008] 24a60004 addiu $6, $5, 4 ; addiu $a2, $a1, 4 #
  envp
- [004000c] 0041080 sll $2, $4, 2 ; sll $v0, $a0, 2
- [00400010] 00c23021 addu $6, $6, $2 ; addu $a2, $a2, $v0
- [00400014] 0c000000 jal 0x00400020 [main]; jal main
- [00400018] 3402000a ori $2, $0, 10 ; li $v0 10
- [0040001c] 0000000c syscall ; syscall

- Imagine doing disk I/O directly:
  - disk description
  - controlling Heads; timing
  - keeping track of where things are stored
  - dealing with Errors

- Other I/O
  - Keyboard
  - Screen
  - Communications

- Keeping track of memory (RAM)
  - Multiple tasks
  - Multiple users

- Sharing the CPU
  - Multiple tasks
  - Multiple users

- The User Interface Problem
  - For the computer professional only?
  - For the lay person
  - The Graphical User Interface
    - Computation to support this?

Historical Perspective

- Early Years
  - Early 1960's machines: Almost Bare
  - Mid 1960's Machines: Early Batch Operating Systems
  - Multiprogramming Systems
  - Time Sharing
  - Lab Computers

- Had Major Theme: CPU Time Precious
  - Ease of use: very low priority
  - Graphical User Interface too costly (and not yet invented)
  - This perspective faded with time and began to disappear with advent of the microprocessor: Cheap CPU time.
Historical Perspective

- **Later Years**
  - Microprocessor in late 70's
  - PC's in early 80's: Operating Systems for PC
  - Apple: Macintosh (Xerox PARC)
  - Workstations
    - UNIX -- AT&T: License Wars -- LINUX
- **Major Change: Lower Costs**
  - Whole new audience
  - User Interface
    - Essential for non pros
    - Affordable (cpu cycles to burn)
  - Whole new competitive environment
  - Volume!

Role of the Operating System

1. **Processor Management (Multiprogramming = sharing)**
   1. Several virtual machines
2. **I/O Systems**
   1. Windowing Systems / GUIs
   2. File Systems (use of your hard disk)
   3. Communications/Networking
3. **Memory (RAM) Management**
   1. Sharing Memory
   2. Simulating Additional Memory (Virtual Memory)
4. **Software Environments**
   - Administration/Accounting
   - Compilers
   - Tools

Memory Management

- **Virtual Memory**
  - Simulate memory using disks
- **Cache Memory**
  - Simulate faster memory using large slow and small fast memory
  - Library, bookcase, desktop analogies
- **Memory Hierarchies**
  - Registers x1
  - Cache x10 - x100
  - Main Memory x100 - x1000
  - Disks x1,000,000
- **Overhead**
  - Card Catalog analogies
  - Finding stuff on your desk or bookcase

Memory Management.2

- **Historically**
  - Swapping in Time Sharing Systems
  - Whole user image involved
- **Paging**
  - Page is conveniently sized block of memory (RAM)
    - (power of 2)
  - Physical swapping done page at a time
- **Protection**
  - Security (write protect)
  - Confidentiality (read protect)
  - (lacking on Early machines)
I/O Systems

- Communications/Networking
  - Extremely important in modern systems
  - (Deal with that before)
- Graphical User Interfaces (GUI)
  - X-Windows
  - Macintosh Desktop
  - MS Windows
  - Born in Xerox PARC; Legal Fun
- Files Systems
  - Flat
  - Hierarchical (Directories/Folders)
  - Distributed Files Systems
    - Andrew File System (AFS) (e.g., at OIT)
    - Network File System (NFS) (e.g., in Computer Science)

Processor Management

- Virtual Machines
  - True Parallel Processes vs. Simulated Parallel
    - Note that the “interleaving” is unpredictable
  - Interrupts
    - Contrast with “busy waiting”
- Process Management
  - Fairness
  - Responsiveness
- Synchronization Problems
  - Danger of shared resources
    - Data: Race conditions
    - Any Exclusive Resource: Deadlocks

Synchronization Problems

- Race Conditions
  - Two Processes (A and B)
    - A manages (updates, etc.) “clock”
    - B uses “clock”
  - Example: clock at 8:59
    - A: add one to minutes – 8:00 – note carry!
    - B: reads clock 8:00!
    - A: add carry to hours 9:00
  - Due to bad timing, B gets a time almost 1 hour off!
- How can we avoid Race Conditions?
- Deadlocks
  - User A needs printer P and modem M
  - User B needs modem M and printer P
  - Both are competing for same resources
  - 3 scenarios possible

Synchronization Problems

<table>
<thead>
<tr>
<th>Scenario 1</th>
<th>Scenario 2</th>
<th>Scenario 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: get P</td>
<td>B: get M</td>
<td>A: get P</td>
</tr>
<tr>
<td>A: get M</td>
<td>B: get P</td>
<td>B: get M</td>
</tr>
<tr>
<td>B: get M (wait!)</td>
<td>A: get P (wait!)</td>
<td>A: get M (wait!)</td>
</tr>
<tr>
<td>A: process</td>
<td>B: process</td>
<td>B: get P (wait!)</td>
</tr>
<tr>
<td>A: release M, P</td>
<td>B: release P, M</td>
<td>...keep waiting...</td>
</tr>
<tr>
<td>B: get M</td>
<td>A: get P</td>
<td>B: get M</td>
</tr>
<tr>
<td>B: get P</td>
<td>A: get M</td>
<td>DEADLOCK!</td>
</tr>
<tr>
<td>B: process</td>
<td>A: process</td>
<td></td>
</tr>
</tbody>
</table>

- How can we avoid Deadlocks?