CSc 360
Operating Systems
Overview

Jianping Pan

Summer 2006

Historical view

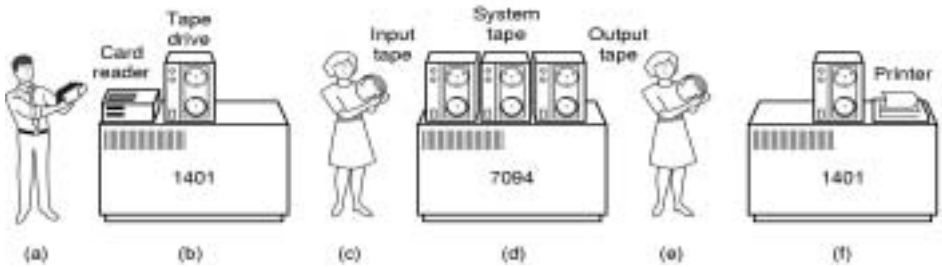
- Was
  - computers were more expensive than users
  - goals: make computers more efficiently used
  - results: share computers
- Now
  - users become more ″expensive″ than computers
  - goals: make computers more effectively used
  - results: share users (among many computers)

Multiprogramming

- ″One program at a time″
  - start, execute, {wait, execute}*, finish
  - wait for: input/output, other programs, etc
  - CPU may be idle most of the time
- Multiprogramming
  - try to keep CPU always busy
  - handle multiple programs at the same time
  - ″share″ CPU

Batch processing

- Load a pool of jobs
- Execute one job *until* it is blocked
- Pick another one to execute



Time sharing

- Execute one job up to a certain time
  - e.g., hardware timer with counter
- Switch to another one to execute
  - job scheduling, memory swapping
- Seem to execute *many* jobs at the same time
- Batch processing vs time sharing
  - job responsiveness
  - swapping overhead

OS operations

- Interrupt the current job
  - *yield*: system call trap (e.g., I/O)
  - *yank*: hardware timer interrupt
  - how about an ″abusive″ job?
- Dual-mode operation
  - user mode for regular applications
  - kernel mode with privileged instructions
  - trap: user to kernel entry

Process management

- Process: a running program
- Create, delete, suspend and resume process
  - resource allocation: CPU, memory, I/O, etc
- Schedule processes/threads
- Synchronize processes
- Communicate between processes
- Handle deadlocks

Memory management

- (Main) memory
  - store instructions for execution
  - store data for processing
- Keep track available memory
- Allocate and reclaim memory
  - provide protected access
  - trap invalid access
- Swap in/out memory

Storage and I/O management

- "In Unix, everything is a file"
  - a uniform, logical interface: open, read, write
- Create and delete files and directories
  - directory is a special file
  - file system hierarchy
- Manipulate files and directories
  - provide protected access
  - handle device-specific issues (disks, I/O, etc)

Cache management

- Memory/storage hierarchy
  - register, L2 cache, main memory, disk, tape, etc
  - trade-off: size vs speed
- Cache operations
- Cache replacement
  - improve cache "hit" ratio
- Consistency control
  - maintain consistency among multiple copies

User management

- Authentication
  - who's who
  - user credentials (e.g., password, token)
- Authorization
  - what can do what
  - access control (e.g., read, write, execute)
- Accounting
  - what has been done (e.g., logging)

Specialized OS

- Different requirements and constraints
  - real-time systems
    - "hard" real-time OS in embedded systems
    - "soft" real-time OS in multimedia systems
  - handheld systems
    - almost a full-blown OS, with resource constraints
  - embedded systems
    - very severe resource constraints

This lecture

- An overview on operating systems
  - multiprogramming
  - process management
  - memory management
  - file management

Next lecture

- Interfaces to OS
  - CLI, GUI, system calls, API
  - read OSC7 Chapter 2 (or OSC6 Chapter 3)