1

CSMC 412

Operating Systems
Prof. Ashok K Agrawala

Set 2

2

Announcements

- Programming Assignment #1 is on the web page
- Reading for next week:
  - Chapter 3 (sections 3.2 to 3.9)
- Programming Assignment #0
  - files are available on the web page

3

Introduction

- What is an operating system?
- Simple Batch Systems
- Multiprogramming Batched Systems
- Time-Sharing Systems
- Personal-Computer Systems
- Parallel Systems
- Distributed Systems
- Real -Time Systems

4

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.
- Use the computer hardware in an efficient manner.

5

Why Study Operating Systems?

- They are large and complex programs
  - good software engineering examples
- There is no perfect OS
  - too many types of users
    - » real-time, desktop, server, etc...
  - many different models and abstractions are possible
    - » OS researchers have been termed abstraction merchants
- Many levels of abstraction
  - hardware details: where the bits really go and when
  - high level concepts: deadlock, synchronization

6

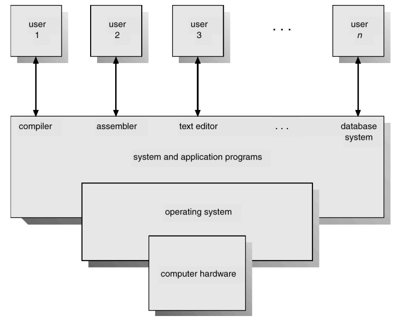
Why Study Operating Systems (cont.)

- Necessity
  - reliability: when the OS is down, computer is down
  - recovery: when the OS goes down it should not take all of your files with it.
- It’s fun
  - the details are interesting (at least I think so :)
  - thinking about concurrency makes you better at writing software for other areas

Computer System Components

1. Hardware – provides basic computing resources (CPU, memory, I/O devices).
2. Operating system – controls and coordinates the use of the hardware among the various application programs for the various users.
3. Applications programs – define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business programs).
4. Users (people, machines, other computers).

Abstract View of System Components



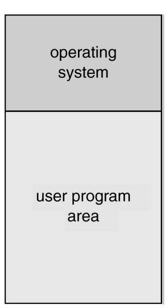
Operating System Definitions

- Resource allocator – manages and allocates resources.
- Control program – controls the execution of user programs and operations of I/O devices .
- Kernel – the one program running at all times (all else being application programs).

Simple Batch Systems

- Hire an operator
- User ≠ operator
- Add a card reader
- Reduce setup time by batching similar jobs
- Automatic job sequencing – automatically transfers control from one job to another. First rudimentary operating system.
- Resident monitor
  - initial control in monitor
  - control transfers to job
  - when job completes control transfers pack to monitor

Memory Layout for a Simple Batch System



Control Cards

- Problems
  1. How does the monitor know about the nature of the job (e.g., Fortran versus Assembly) or which program to execute?
  2. How does the monitor distinguish
     (a) job from job?
     (b) data from program?
- Solution
  - Introduce control cards

Control Cards (Cont.)

- Special cards that tell the resident monitor which programs to run
  $JOB
  $FTN
  $RUN
  $DATA
  $END
- Special characters distinguish control cards from data or program cards:
  $ in column 1
  // in column 1 and 2
  709 in column1

Control Cards (Cont.)

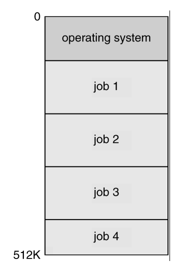
- Parts of resident monitor
  - Control card interpreter – responsible for reading and carrying out instructions on the cards.
  - Loader – loads systems programs and applications programs into memory.
  - Device drivers – know special characteristics and properties for each of the system's I/O devices.
- Problem: Slow Performance – I/O and CPU could not overlap ; card reader very slow.
- Solution: Off-line operation – speed up computation by loading jobs into memory from tapes and card reading and line printing done off-line.

Spooling

- Overlap I/O of one job with computation of another job. While executing one job, the OS.
  - Reads next job from card reader into a storage area on the disk (job queue).
  - Outputs printout of previous job from disk to printer.
- *Job pool* – data structure that allows the OS to select which job to run next in order to increase CPU utilization.

Multiprogrammed Batch Systems

Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.



OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.

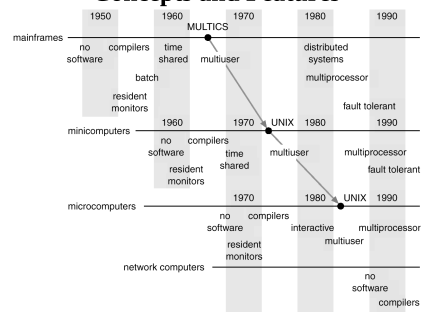
Time-Sharing Systems– Interactive Computing

- The CPU is multiplexed among several jobs that are kept in memory and on disk (the CPU is allocated to a job only if the job is in memory).
- A job swapped in and out of memory to the disk.
- On-line communication between the user and the system is provided; when the operating system finishes the execution of one command, it seeks the next "control statement" not from a card reader, but rather from the user's keyboard.
- On-line system must be available for users to access data and code.

Personal-Computer Systems

- *Personal computers* – computer system dedicated to a single user.
- I/O devices – keyboards, mice, display screens, small printers.
- User convenience and responsiveness.
- Can adopt technology developed for larger operating system' often individuals have sole use of computer and do not need advanced CPU utilization of protection features.

Migration of Operating-System Concepts and Features



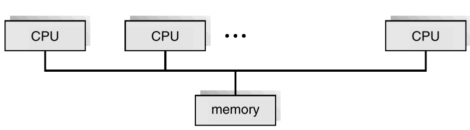
Parallel Systems

- Multiprocessor systems with more than on CPU in close communication.
- *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.
- Advantages of parallel system:
  - Increased *throughput*
  - Economical
  - Increased reliability
    - » graceful degradation
    - » fail-soft systems

Parallel Systems (Cont.)

- *Symmetric multiprocessing (SMP)*
  - Each processor runs and identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems support SMP
- *Asymmetric multiprocessing*
  - Each processor is assigned a specific task; master processor schedules and allocated work to slave processors.
  - More common in extremely large systems

Symmetric Multiprocessing Architecture



Real-Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- Well-defined fixed-time constraints.
- *Hard real-time system*.
  - Secondary storage limited or absent, data stored in short-term memory, or read-only memory (ROM)
  - Conflicts with time-sharing systems, not supported by general-purpose operating systems.
- *Soft real-time system*
  - Limited utility in industrial control or robotics
  - Useful in applications (multimedia, virtual reality) requiring advanced operating-system features.

Distributed Systems

- Distribute the computation among several physical processors.
- *Loosely coupled system* – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- Advantages of distributed systems.
  - Resources Sharing
  - Computation speed up – load sharing
  - Reliability
  - Communications

Distributed Systems (Cont.)

- Network Operating System
  - provides file sharing
  - provides communication scheme
  - runs independently from other computers on the network
- Distributed Operating System
  - less autonomy between computers
  - gives the impression there is a single operating system controlling the network.