

## Chapter 1: 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

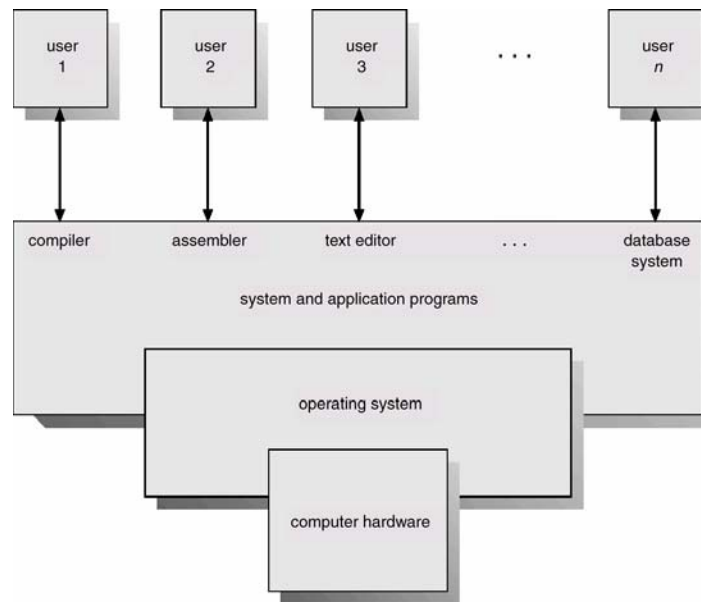
## What is an Operating System?

- 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.

## 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  $\neq$  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 back to monitor

## Memory Layout for a Simple Batch System



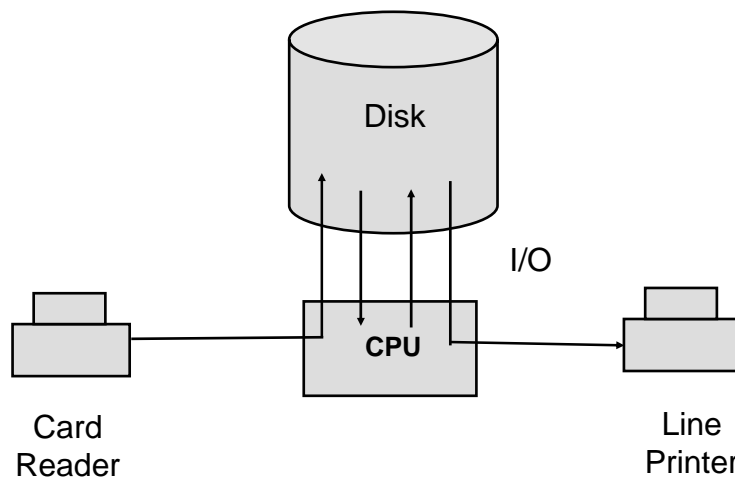
## Control Cards

- 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.)

- **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.
- In execution environment, the CPU is often idle, because the speeds of the I/O devices are slower than of electronic devices (CPU).
- A solution for this problem is spooling.

## Spooling

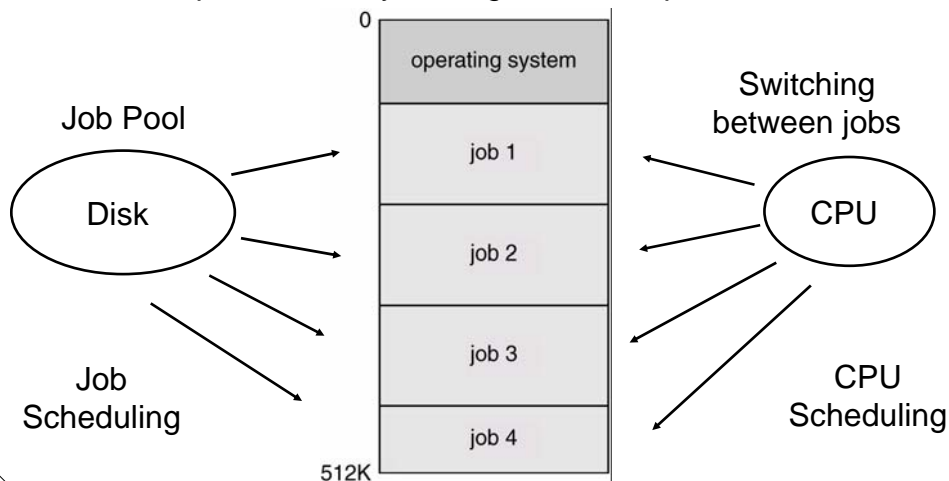


## Spooling

- Overlap I/O of one job with computation of another job. While executing one job, the O.S.
  - 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.
- The spooler may be reading the input of one job while printing the output of a different job.
- Spooling can keep both the CPU and the I/O devices working at much higher rates.
- In Spooling, jobs must be run sequentially, first-come first-served.

## Multiprogramming Batch Systems

Multiprogramming: several jobs are kept in main memory at the same time, and the CPU is multiplexed among them which requires memory management and protection.



## Multiprogramming Batch Systems

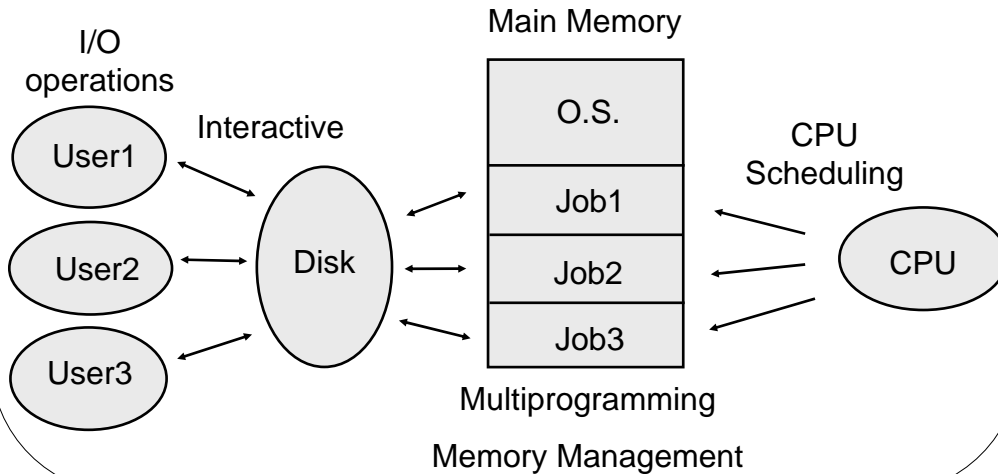
- The O.S. picks and begins to execute one job from memory. Once this job needs an I/O operation the O.S. switches to another job (CPU or O.S. always busy).
- The number of jobs in memory is less than the number of jobs in disk (Job Pool).
- If several jobs are ready to be brought into memory and there is not enough room for all of them, then the system choose jobs among them (Job Scheduling).
- If several jobs are ready to run at the same time, the system must choose among them (CPU Scheduling).
- Having several programs in memory at the same time requires memory management.
- In non-multiprogrammed system, CPU sit idle.
- In multiprogramming system, CPU will never be idle.

## OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
  - Memory management – the system must allocate the memory to several jobs.
  - Job scheduling – the system must choose among several jobs ready to run from Disk.
  - CPU scheduling – the system must choose among several jobs ready to run in memory.
  - Allocation of devices.
- 
- Two main disadvantages of Multiprogrammed Batched Systems:
    - Users cannot interact with their jobs, while executing.
    - A programmer cannot modify a program as it executes to study its behavior.

## Time-Sharing Systems – Interactive Computing

A time-sharing system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer.



Operating System Concepts

1.15

## 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).
- When a job needs an I/O operation, the CPU switches between jobs. Therefore, the CPU is always busy.
- A job is swapped in and out of memory to the disk which serves as a back up for main memory.
- 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.

Operating System Concepts

1.16

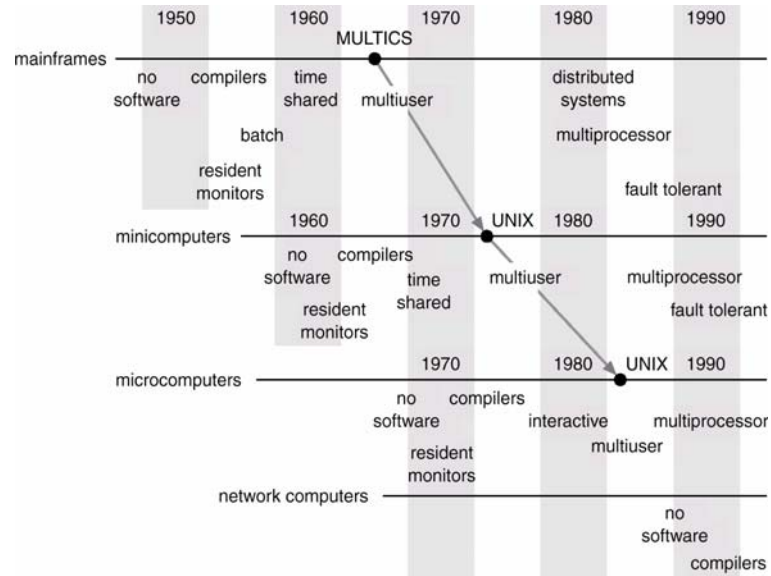
## **Time-Sharing Systems (Cont.)**

- Time-Sharing Systems provide the following:
  - On-Line file system, where the files are on a collection of disks. Therefore, disk management must be provided.
  - A mechanism for concurrent execution, which requires CPU scheduling schemes.
  - Mechanisms for job synchronization and communication to ensure orderly execution.
  - A mechanism to avoid deadlock - a job waiting for another job forever.

## **Personal-Computer Systems**

- Personal computers – computer system dedicated to a single user.
- I/O devices – keyboards, mice, display screens, small printers.
- PC operating systems were neither multi-user nor multi-tasking.
- The goal of PC operating systems were to maximize user convenience and responsiveness instead of maximizing CPU and I/O utilization.
- Examples: Microsoft Windows and Apple Macintosh

## Migration of Operating-System Concepts and Features

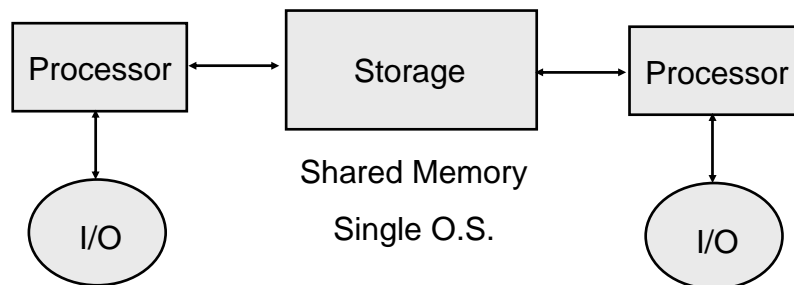


Operating System Concepts

1.19

## Parallel Systems

- Multiprocessor systems with more than one CPU in close communication.
- *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.



Operating System Concepts

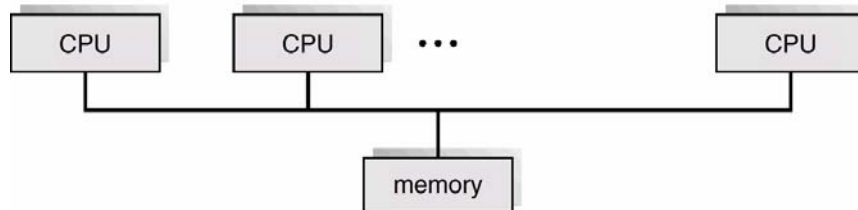
1.20

## Parallel Systems (Cont.)

- Advantages of parallel system (multiprocessor systems):
  - Increased *throughput* – number of processes that are completed per time unit.
  - *Economical* (for large jobs) - no need to make copies of data and distribute it among several processors.
  - *Faster* (for large jobs) – divide the work on all processors.
  - Increased *reliability* (fault – tolerant) – For example, if we have 10 processors working together on a job and one processor failed, then the remaining 9 processors must pick up a share of the work of the failed processor. Thus, the entire system is still working but slower by 10%. Therefore, multiprocessor systems are reliable.

## Parallel Systems (Cont.)

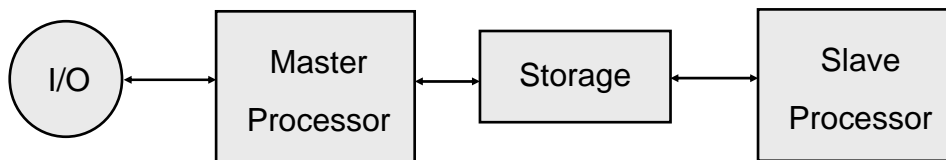
- *Symmetric multiprocessing (SMP) model*
  - Each processor runs an identical copy of the operating system.
  - Most modern operating systems support SMP, such as UNIX for Multimax computer.



Symmetric Multiprocessing Architecture

## Parallel Systems (Cont.)

- *Asymmetric multiprocessing model*
  - Each processor is assigned a specific task; master processor schedules and allocates work to slave processors.



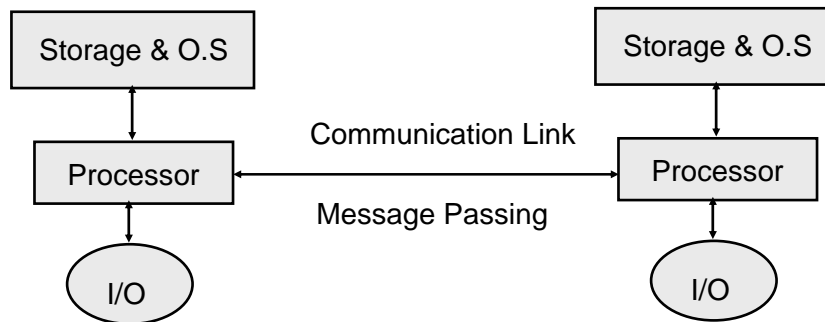
Asymmetric Multiprocessing Architecture

## Parallel Systems (Cont.)

- *Asymmetric multiprocessing model (Cont.):*
  - Master performs I/O and computations.
  - Only master may execute the O.S.
  - Slave can execute only user programs.
  - If master fails the system cannot perform I/O.
  - If slave fails some computations are lost but still the system can function.
  - More common in extremely large systems.

## Distributed Systems

- Distribute the computation among several physical processors.
- *Loosely coupled system* – involves connecting 2 or more independent computer systems via communication link. So, each processor has its own O.S. and local memory; processors communicate with one another through various communications lines (message passing), such as high-speed buses or telephone lines.



Operating System Concepts

1.25

## Distributed Systems (Cont.)

- Advantages of distributed systems:
  - Resources Sharing – You can share files and printers.
  - Computation speed up – A job can be partitioned so that each processor can do a portion concurrently (load sharing).
  - Reliability – If one processor failed the rest still can function with no problem.
  - Communications – Such as electronic mail, ftp, etc.

Operating System Concepts

1.26

## Real-Time Systems

- Real – Time Systems are characterized by supplying immediate response. For example, sensors bring data to the computer.
- 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.