**Chapter 1**
Introduction and
Fundamental Concepts

# OPERATING SYSTEMS
Design and Implementation

**Instructor:**
Hadi Salimi
Computer Engineering Department
IRAN University of Science and Technology
hsalimi@iust.ac.ir

## Course Resources

- Textbook:
  - Andrew S. Tanenbaum, "Operating Systems, Design and Implementation", (Second Edition), Prentice Hall.
- Supplementary Textbooks:
  - Avi Silberschatz, "Operating System Concepts", 5th Edition, Wiley & Sons, Inc.
  - William Stallings, "Operating Systems: Internals and Design Principles", Prentice Hall, Fifth Edition.
- Course Homepage:
  - http://webpages.iust.ac.ir/hsalimi/???

# Grading Policy

- Homework: 2 (10%)
- Quiz: 2 (10%)
- Project: 3 (15%)
- Midterm: 5 (25%)
- Final: 8 (40%)
- Note:
  - if ((Midterm + Final + Quiz) < 10) then fail!!!

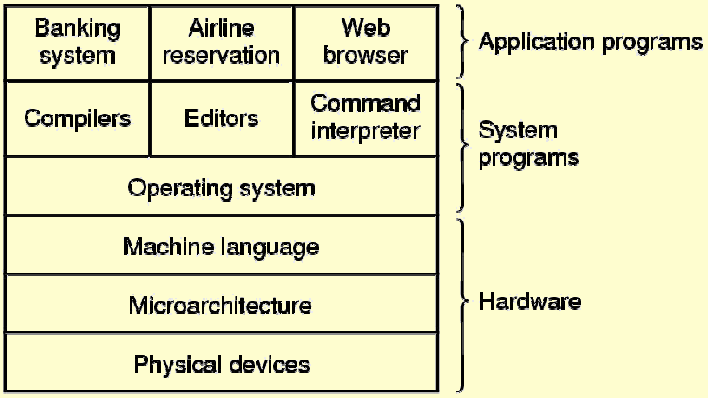


# Chapter Outline

- What is an operating system
- History of operating systems
- Operating system concepts
- System calls
- Operating system structure

# Layers

- Hardware, System Software, Application Software.






# Why an Operating System?

- It is an extended machine
  - Hides the messy details which must be performed
  - Presents user with a virtual machine, easier to use
- It is a resource manager
  - Each program gets time with the resource
  - Each program gets space on the resource

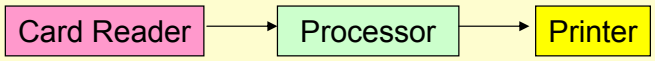
# History

- The First Generation (1945-55)
  - There was no technology higher than vacuum tubes.
  - Programming languages were unknown.
  - Operating System were unheard of.
  - All programming was done in absolute machine language.

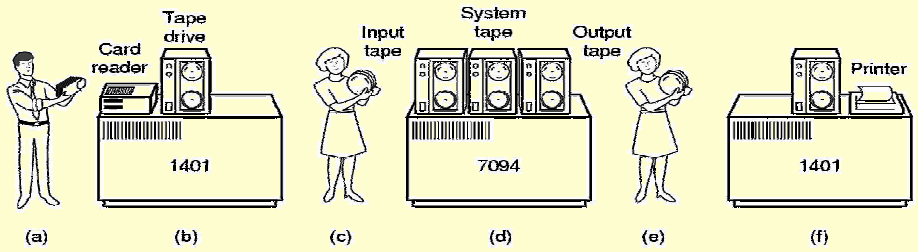


# History (cont.)

- The Second Generation (1955-1965)
  - By introducing the transistor technology, the picture changed rapidly.
  - To run a **job**, the programs were needed to be punched on cards.
  - So, the processor utilization could not increase because of different processing speeds.



# History (cont.)

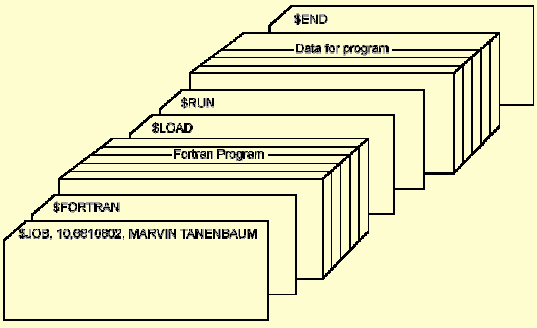



- To avoid that, batch systems were introduced.
  - bringing cards to 1401
  - 1401 reads cards to a tape
  - put tape on 7094 which does computing
  - put tape on 1401 which prints output



# Job Structure

- It started out with $JOB card, specifies job properties.
- The next card tells the OS to load FORTRAN compiler.
- The $LOAD card directs the OS to load the object program just compiled.




- Typical OSs were FMS (Fortran Monitor System) and IBSYS. (IBM's OS for 7094)

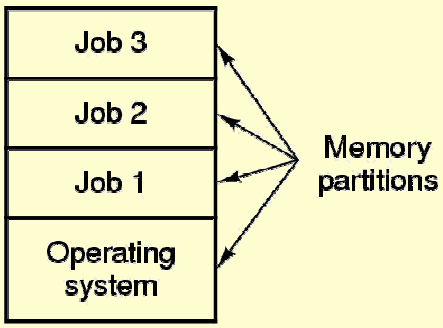
# History (cont.)

- The 3rd generation:
  - Integrated circuits were appeared in this generation.
  - IBM attempted to integrate both 1401 and 7094 into its new System/360.
  - The OS/360 which was developed for this hardware consists of millions of lines of assembly code.
  - **Multiprogramming** was first introduced in it.



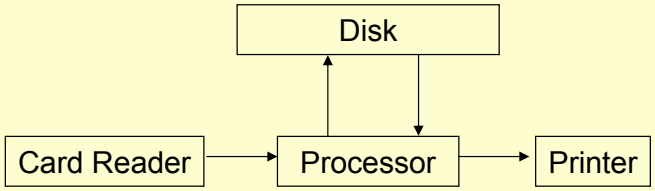
# Multiprogramming

- The solution was to partition the memory into pieces.
- While on job was waiting for I/O another job could use the CPU.
- If enough jobs could be held in the main memory, the CPU could be kept busy nearly 100%.



# History (cont.)

- Another major feature in this generation of operating systems was the ability of reading cards into disk.
- Whenever a running job finished, the OS could load a new job from the disk.
- The technique is called **spooling**. (Simultaneous Peripheral Operation On Line)






# Time Sharing Systems

- Using the third generation systems, the time between submitting a job and getting back the output, was often several hours.
- A single misplaced comma could cause a compilation to fail, and the programmers to waste half a day.
- This desire for quick response time paved the way for **timesharing**.

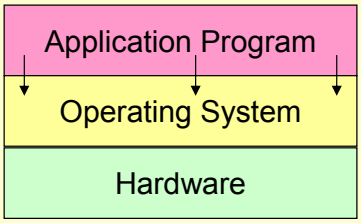
# History (cont.)

- The 4th Generation (1980-now):
  - The age of the personal computer dawned using LSI (Large Scale Integration).
  - Two operating systems initially dominated the PCs: Microsoft's MS-DOS and UNIX.
- Network Operating Systems
- Distributed Operating Systems



# OS Concepts

- **System Call**
  - The interface between the operating system and the user program is defined by a set of "extended instructions".
  - They have been traditionally known as system calls.



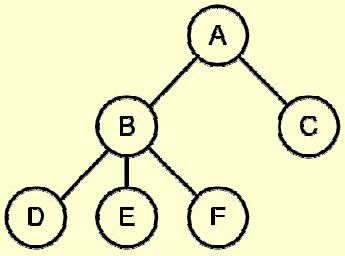
# Processes

- The key concept in all operating systems is the **process**.
- It basically is a program in execution.
- An address space and some set of registers are associated with each process.
- In many operating systems, the information about the processes are storing in a table called **process table**.



# Processes

- The key process management system calls are those dealing with the creation and termination of processes.
- Each process may create one or more child processes.
- Example:
  - A: Shell
  - B: Compiler
  - D: Linker

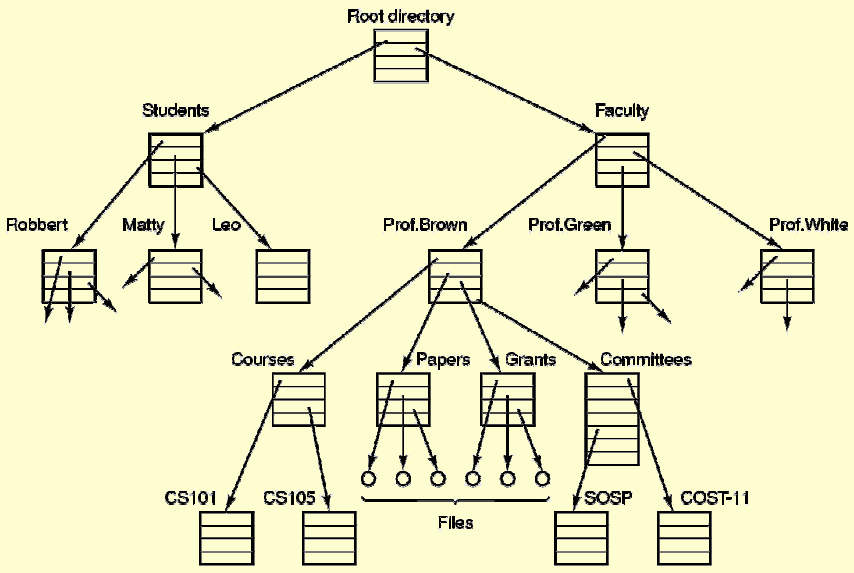


# Files

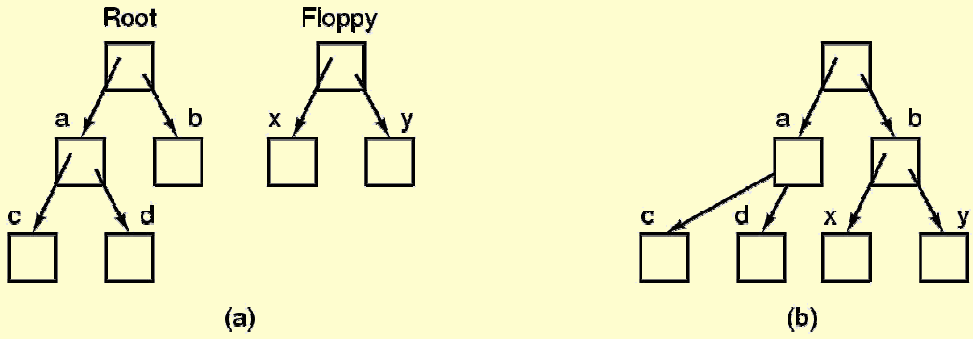
- The other broad category of system calls are related to **files**.
- These system calls handle creation, opening, reading, writing and closing files.
- Other may act as a directory structure handler.



# File System



# Mounting




- Before mounting the files on floppy are inaccessible, but after it the floppy becomes part of the file system.



# Shell

- It makes heavy use of many system call.
- In MINIX, the command interpreter, called the shell, creates a child process for each command that user types.
- $ date
  - The shell creates a child process and runs the date program as the child and waits for it to terminate.

# System Calls and POSIX

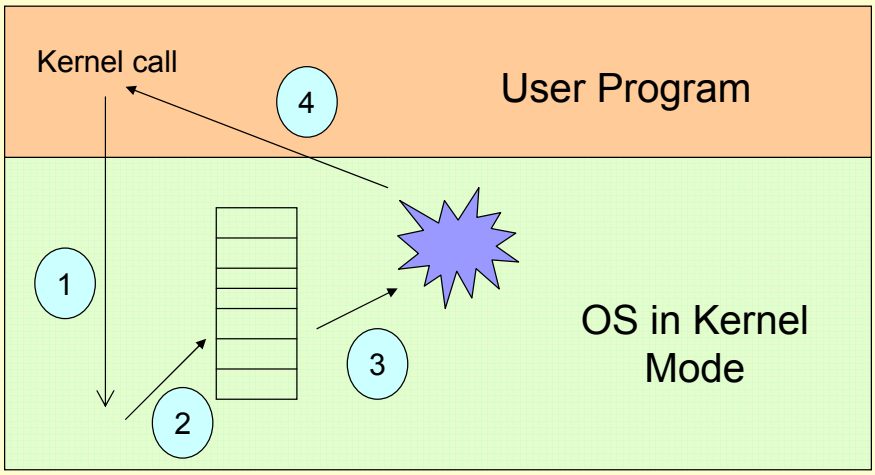
- We can now begin to look at the interface between the operating system and its application programs as the set of system calls.
- To make sure that application programs can run on different operating systems, an international standard, namely POSIX has been specified.



# OS Structure

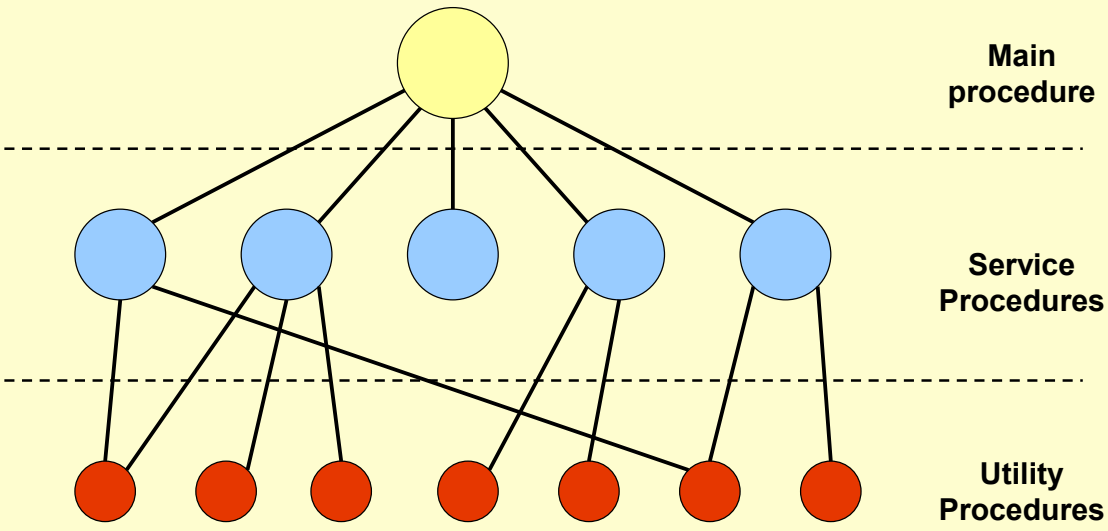
- Monolithic system
- Layered system
- Virtual machine
- Client/Server

# Monolithic system

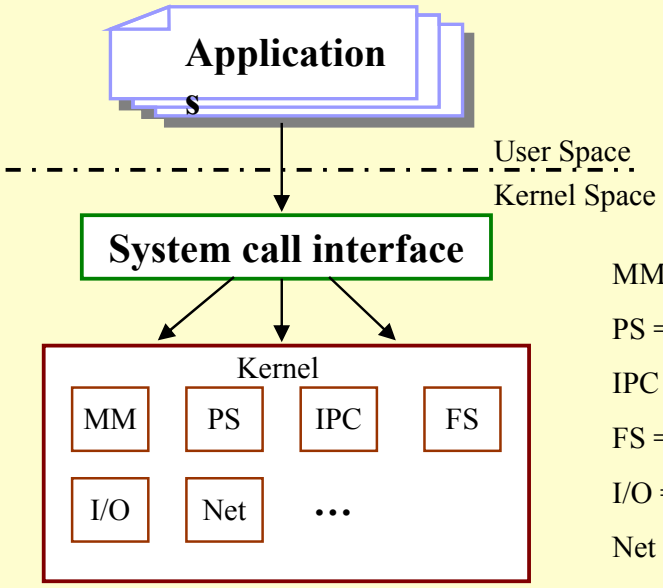





# Monolithic system



# Monolithic system




MM = Memory Manager

PS = Processor Scheduler

IPC = Inter Process Communication

FS = File System

I/O = Input/Output manager

Net = Network manager



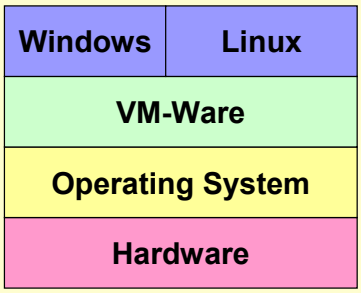
# Layered Systems

| Layer | Function |
|---|---|
| 5 | The operator |
| 4 | User programs |
| 3 | Input/output management |
| 2 | Operator-process communication |
| 1 | Memory and drum management |
| 0 | Processor allocation and multiprogramming |

Structure of THE operating system

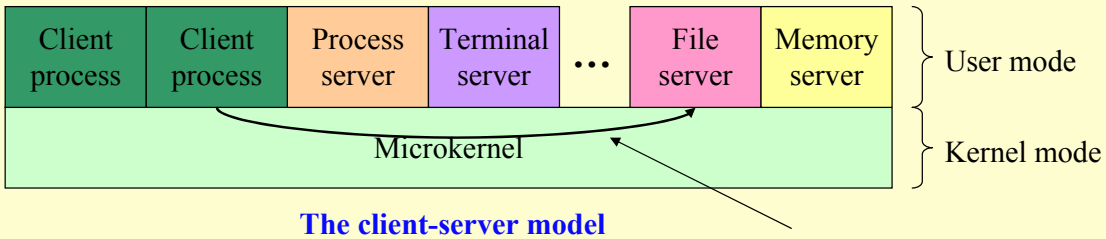
# Virtual Machines

- The idea of virtual machines is heavily used nowadays in different contexts.
- As an example:





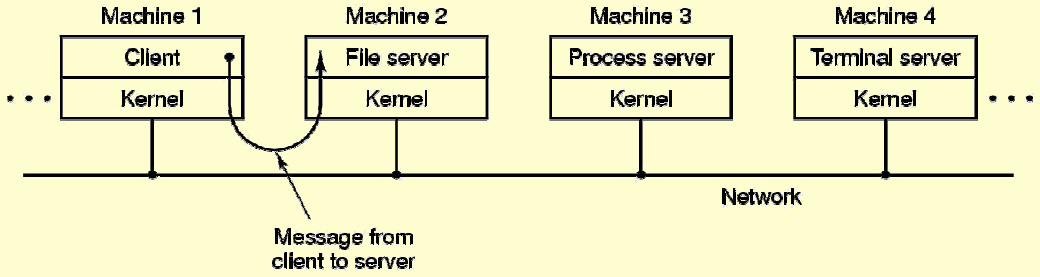

# Client-Server Model




**The client-server model**

- Pros and Cons:
  - Modular, Easily can be distributed, Safer
  - Reduces System Performance

# Client-Server Model




- The client server model in a distributed system.