# Operating Systems
## (*Principles of Operating Systems*)

Đỗ Quốc Huy
huydq@soict.hust.edu.vn
Department of Computer Science
School of Information and Communication Technology

# Course Info

Text book:

- Operating System Concepts – Abraham Siblerschatz

- Modern Operating System – Andrew Tanenbaum

Group Projects

# Chapter 1 Operating System Overview

① Notion of operating system
② History of operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

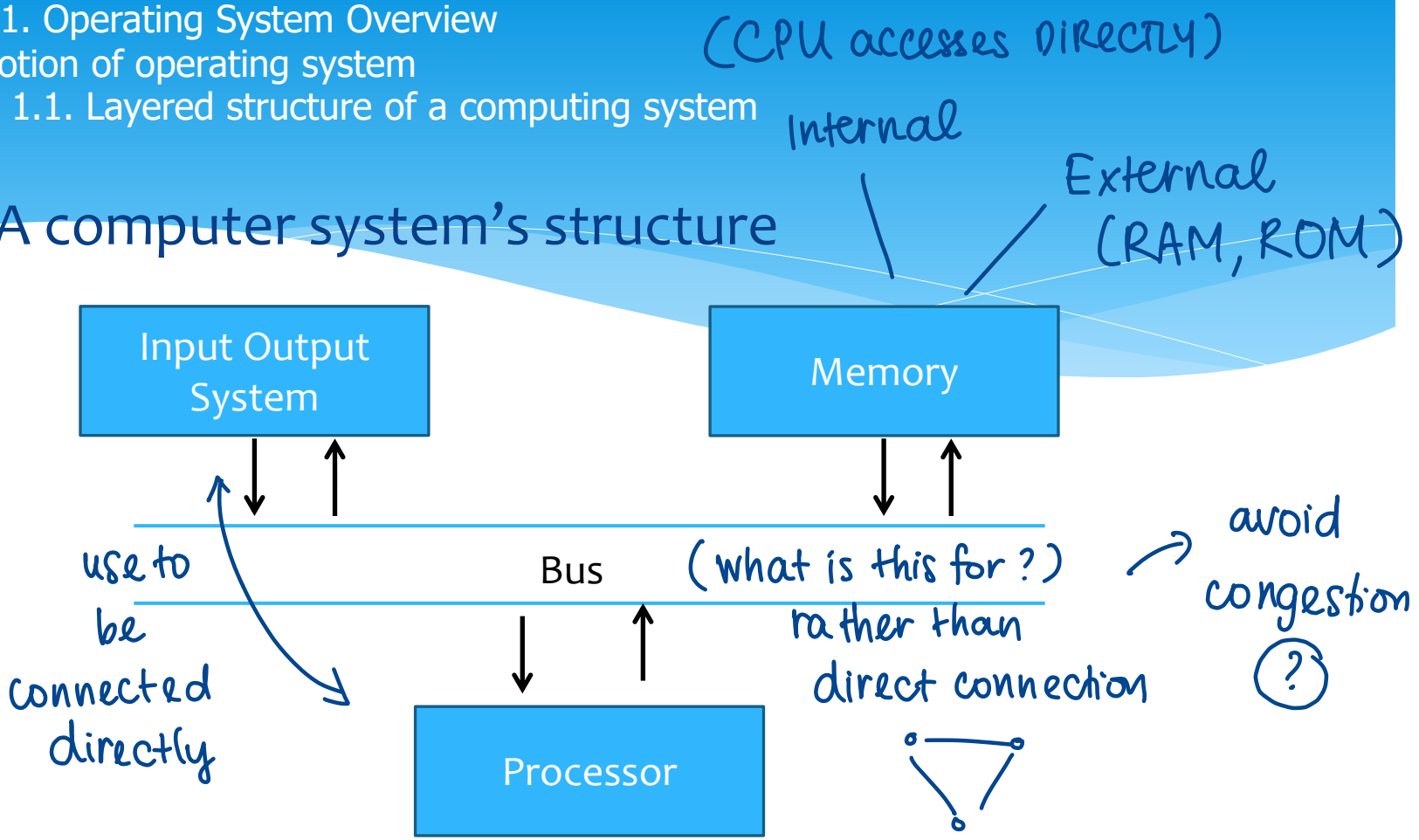
# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

## ① Notion of operating system

- Layered structure of a computing system
- Operating system's functions

- A computer system's structure




- One/ many CPUs, controlling devices are linked by a common bus system to access a shared memory.
- Controlling devices and CPU operate simultaneously and compete with each other.

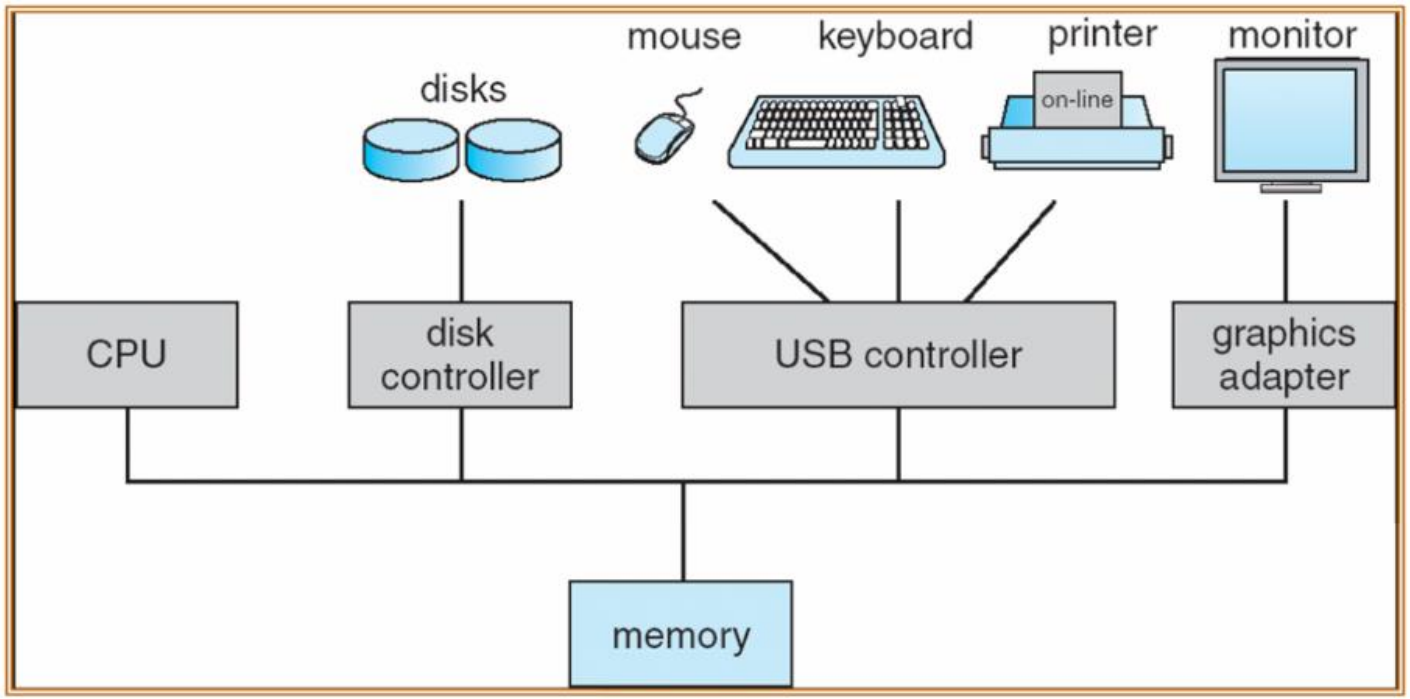
Random access : directly to index

≠

RAM : What does 'Random Access' mean?

↳ Sequential access : traverse from 0 → dest

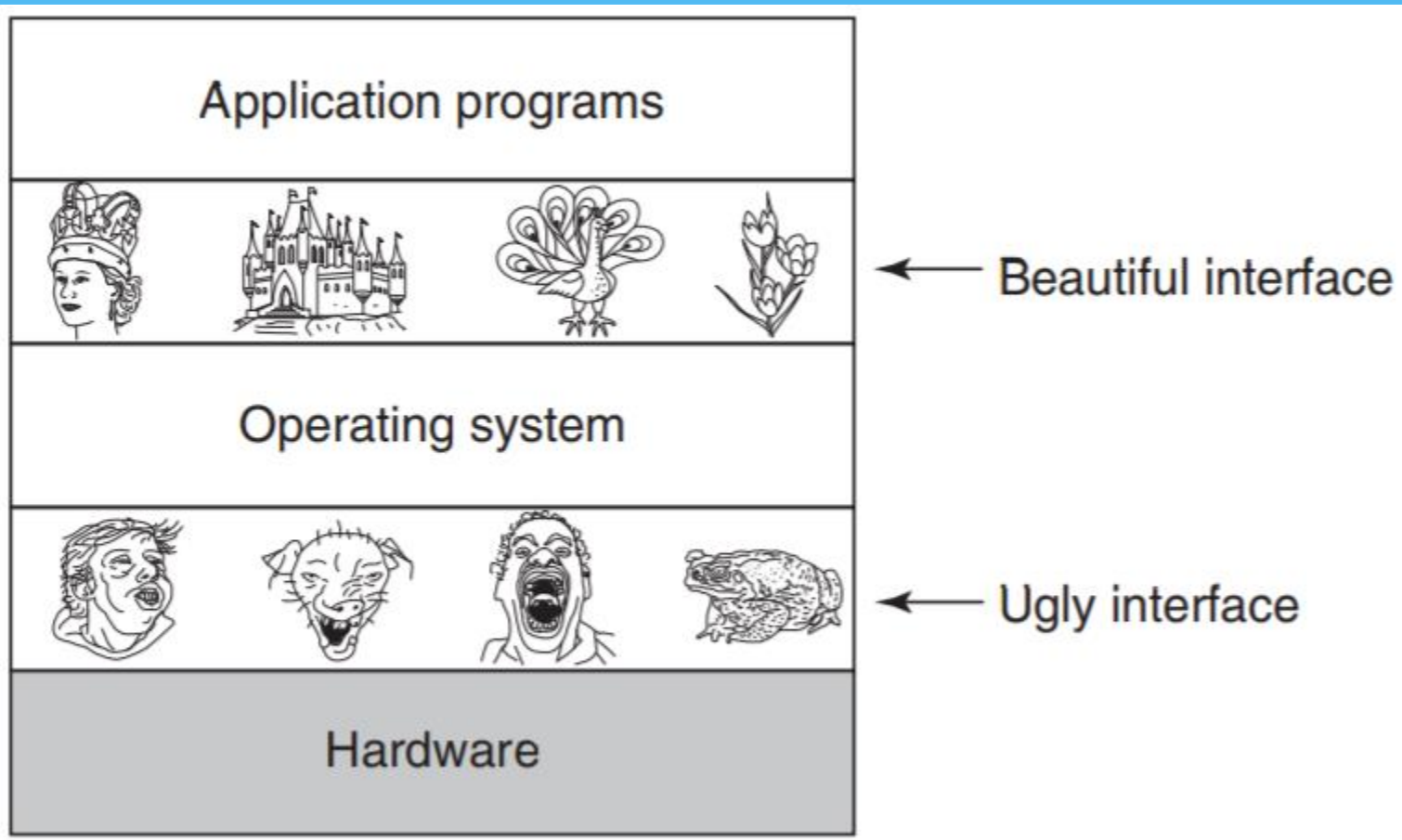

- A computer system's structure



Chapter 1. Operating System Overview

1. Notion of operating system

1.1. Layered structure of a computing system




Operating systems turn ugly hardware into beautiful abstractions

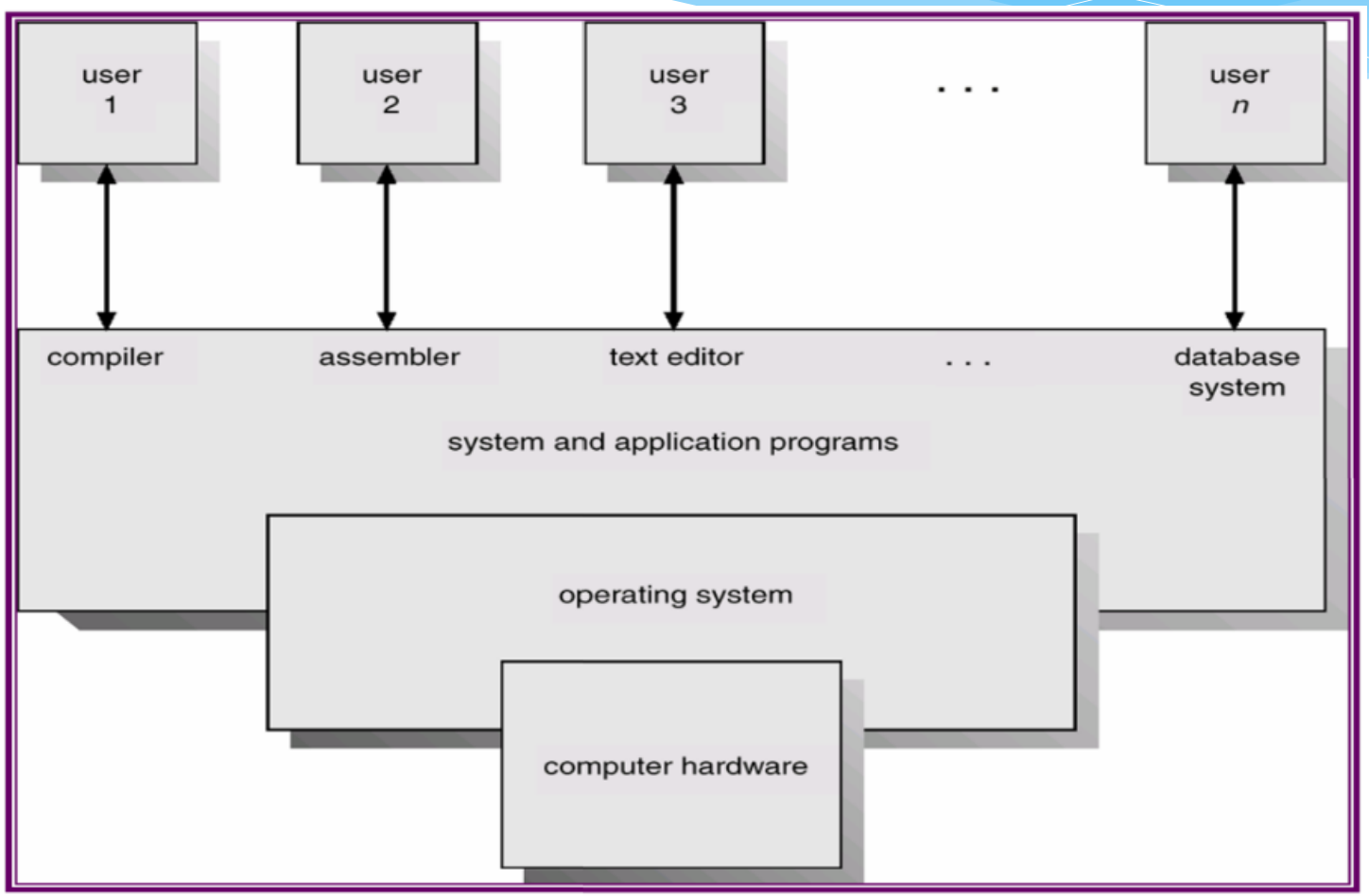
Chapter 1. Operating System Overview

1. Notion of operating system

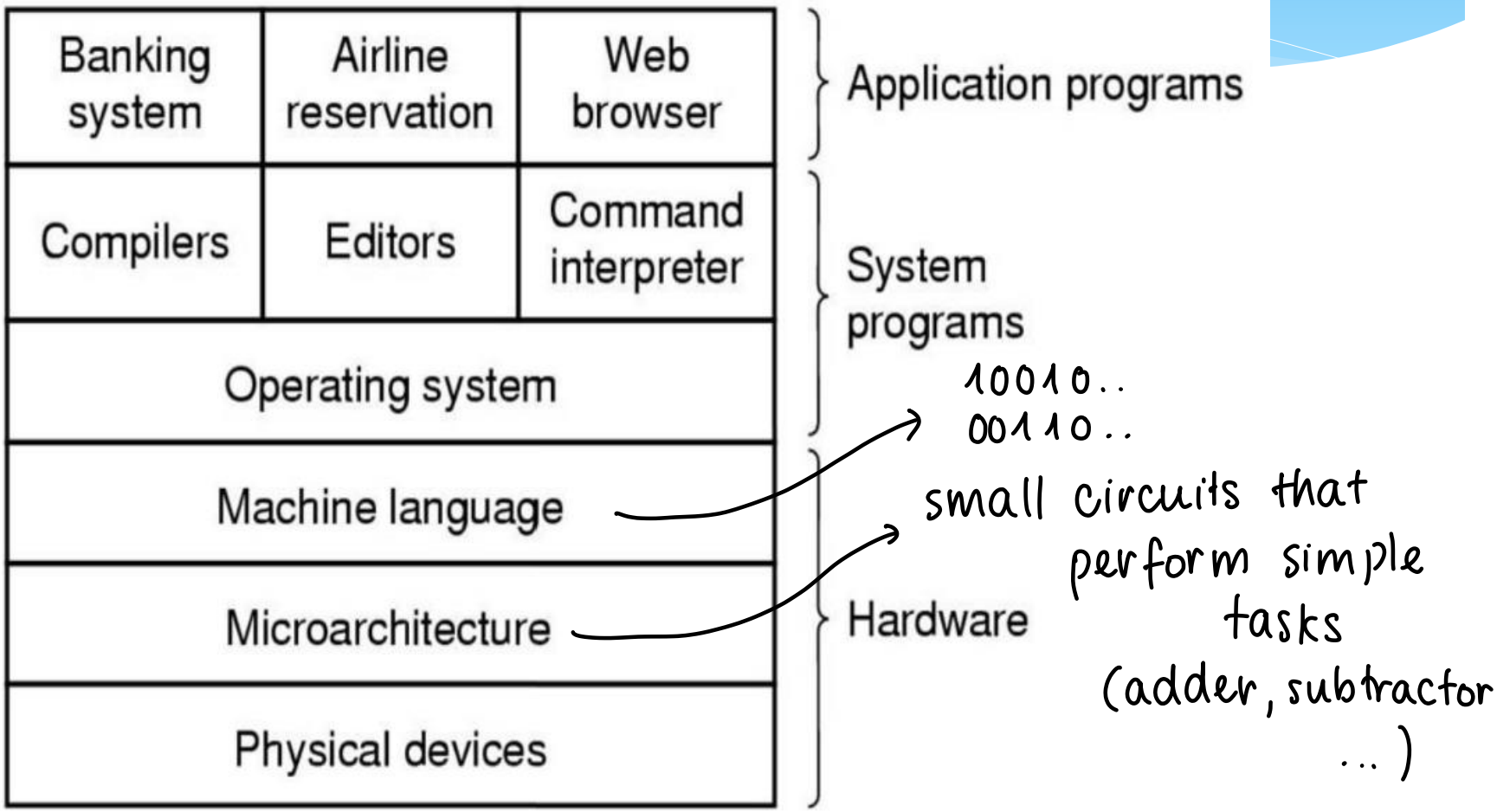
1.1. Layered structure of a computing system

(The architecture of a computing system?)

- A computer system's components (Silberschatz 2002) (1)



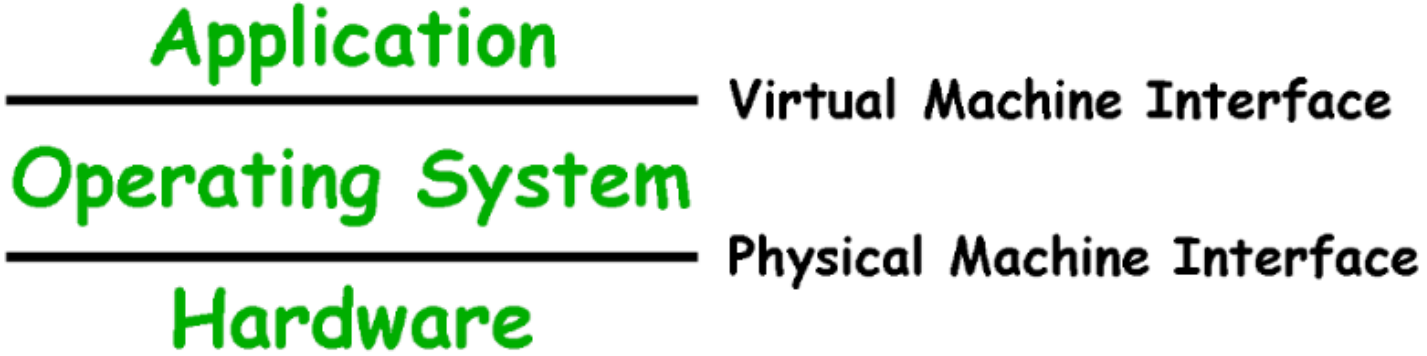
- A computer system's components (Tanenbaum 2001)



- A computing system's components
- Hardware: Provides basic computing resources (CPU, memory, input-output devices)
- Operating system: controls and cooperates hardware using works for application programs of different users.
- Application programs: (compiler, database system, game...) utilizes computer's resource to handle users' requests.
- Users: People who work/operate the machines or computers

(end-users, dev, operators...)

- **Objectives**
- Operating system lies between the system's hardware and application program




- **Objectives**: To provide an environment which helps user run application program and use the computer system easier, more conveniently and effectively.
  - Standardize the user interface for different hardware systems
  - Utilize hardware resource effectively and exploit the hardware performance optimally.

UI looks the same as long as it is the same OS

# ① Notion of operating system

- Layered structure of a computing system
- Operating system's functions

① Simulate a virtual computer machine

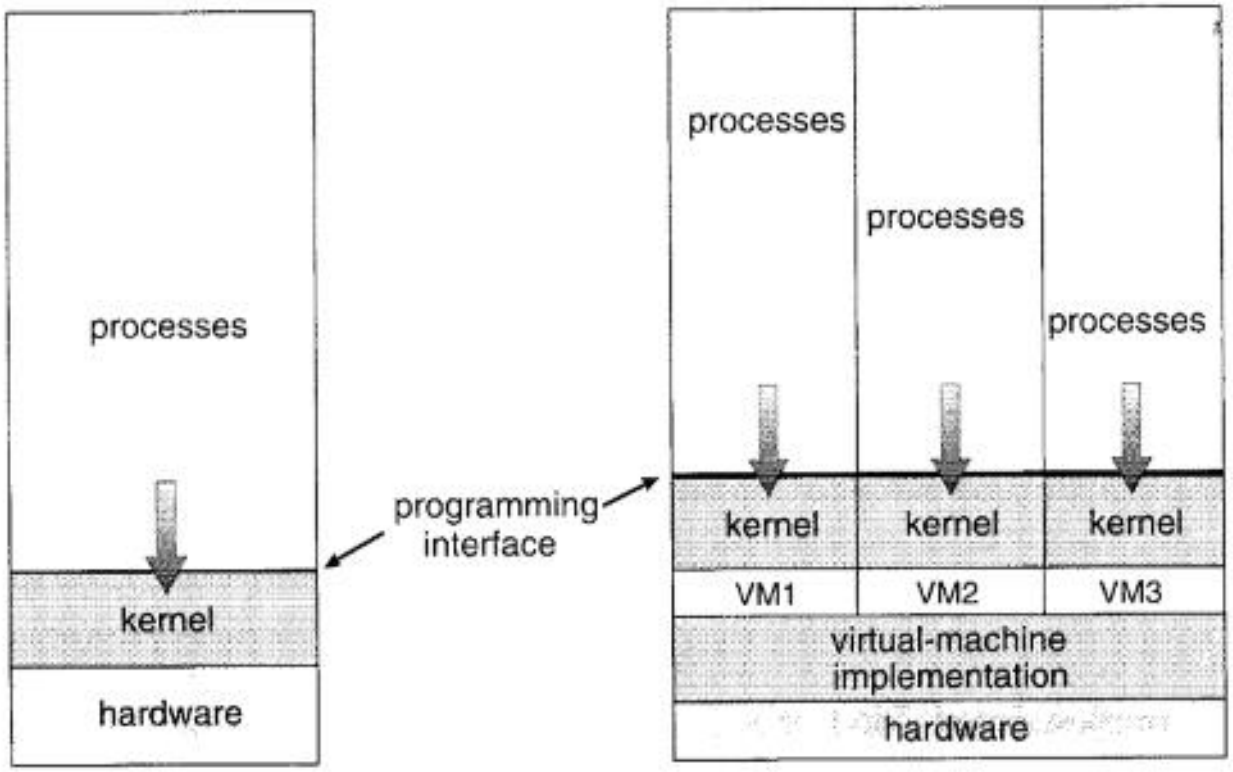
② Manage system resources

# Simulate a virtual computer machine

Help hide detailed works and exploit computer hardware's functions easier and more effectively.

- Simplify programming problem
  - No need to work with binary sequences
  - Each program thinks that it own the whole computer's memory, CPU time, devices...
  - Help communicating with devices easier than with original device. Ex: Ethernet card: Reliable communication, ordered (TCP/IP)
- Extend the system's abilities: The system seem to have desired resources (virtual memory, virtual printer...)
- Help programs not violating each other → a program which does not work would not damage the whole system
- Useful for operating system's development
  - If the experimental operating system get errors, only limited in the virtual machine
  - Help verifying other programs in the operating system

# Simulate a virtual computer machine




Non virtual machine

With virtual machine

# System's resources management

- System's resources (CPU, memory, IO devices, files...) are utilized by program to perform a determined task.
- Programs require resources: time (CPU-usage) and space (memory)
- The operating system has to manage the resource so that the computer can work in the most effective way.
  - Provide resource for program when it's necessary.
  - Competition handling
  - Decide the order of resource providing for the programs' requirements
  - Example: memory resource management (limited)
    - Many program can operate at the same time.
    - Avoid illegal access
    - Data protection (memory sharing: file)

# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

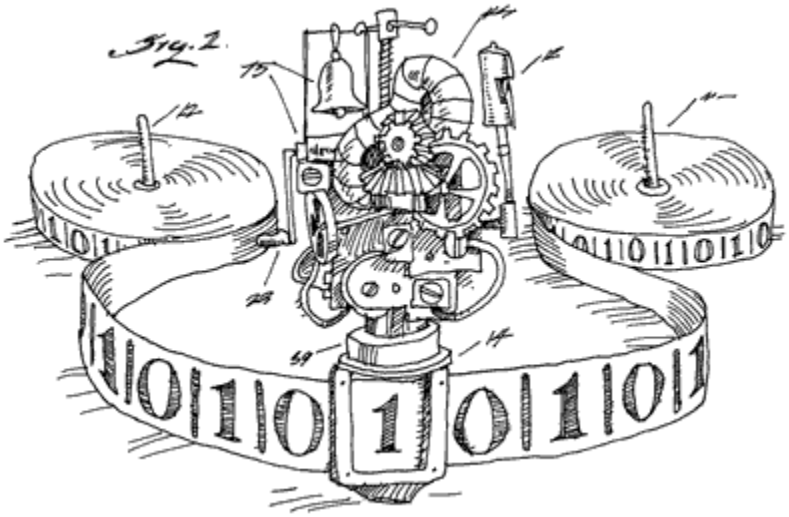
# Operating systems development's history

- History of electronic computer
- Operating systems development's history

# Development history of electronic computers

- 1936 - A. Turing & Church present logic computing model and prove the existence of a computer: Turing machine

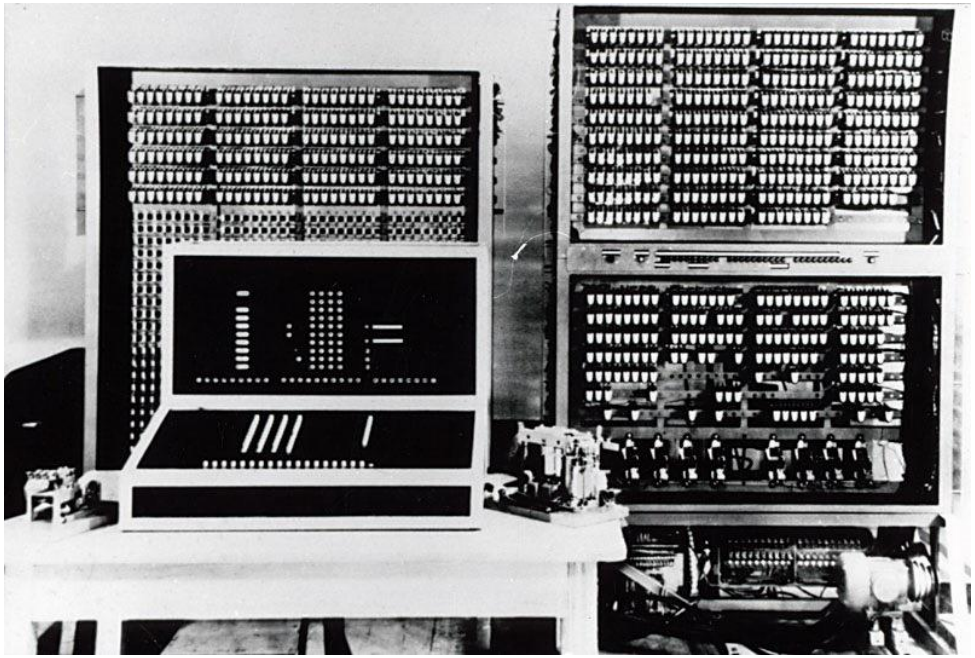
  - Model of a character processing device; Simple but able to perform all the computer's algorithm
  - A Turing machine that is able to simulate any other Turing machine -> a universal Turing machine
  - *Turing is considered as the father of computer science and artificial intelligence*



# Development history of electronic computers

- 1941- Konrad Zuse (German) Constructed world's first programmable computer; the functional program-controlled Turing-complete Z3

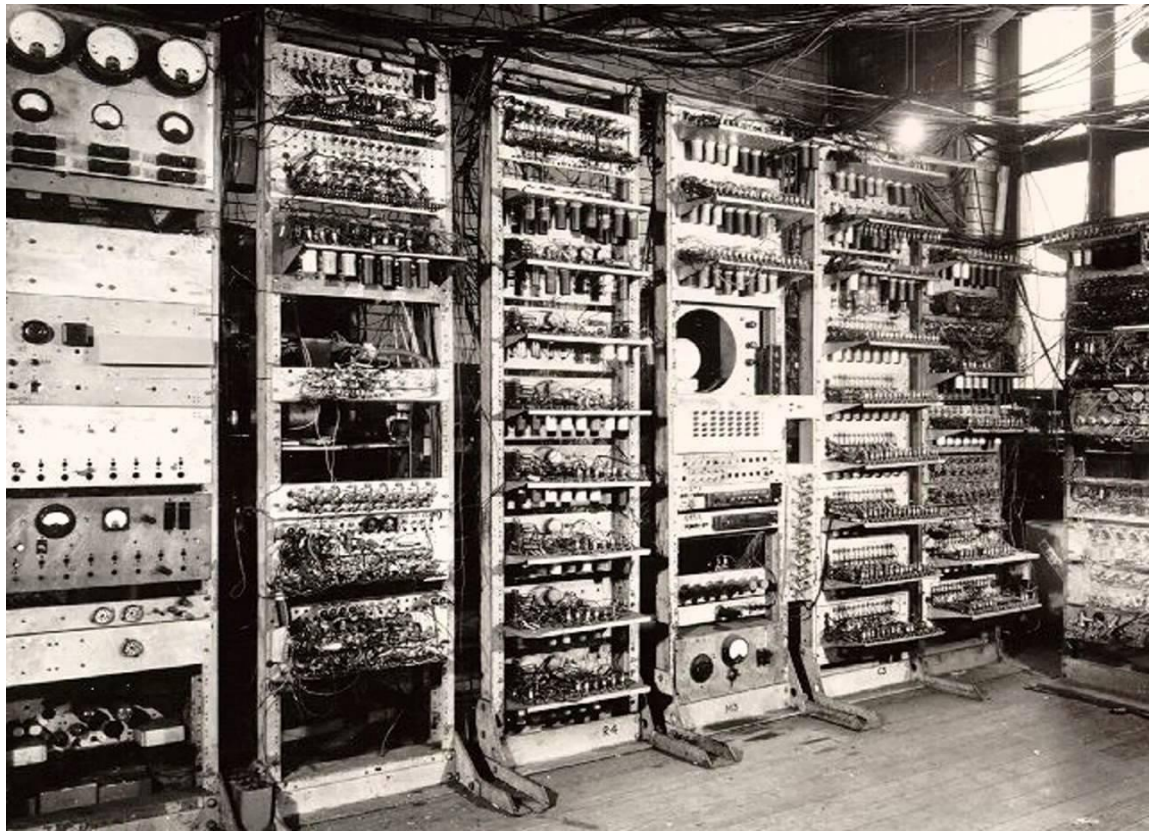
  - Z3: use binary system
  - *Has separated memory and controller*
  - *Mechanical technique*



# Development history of electronic computers

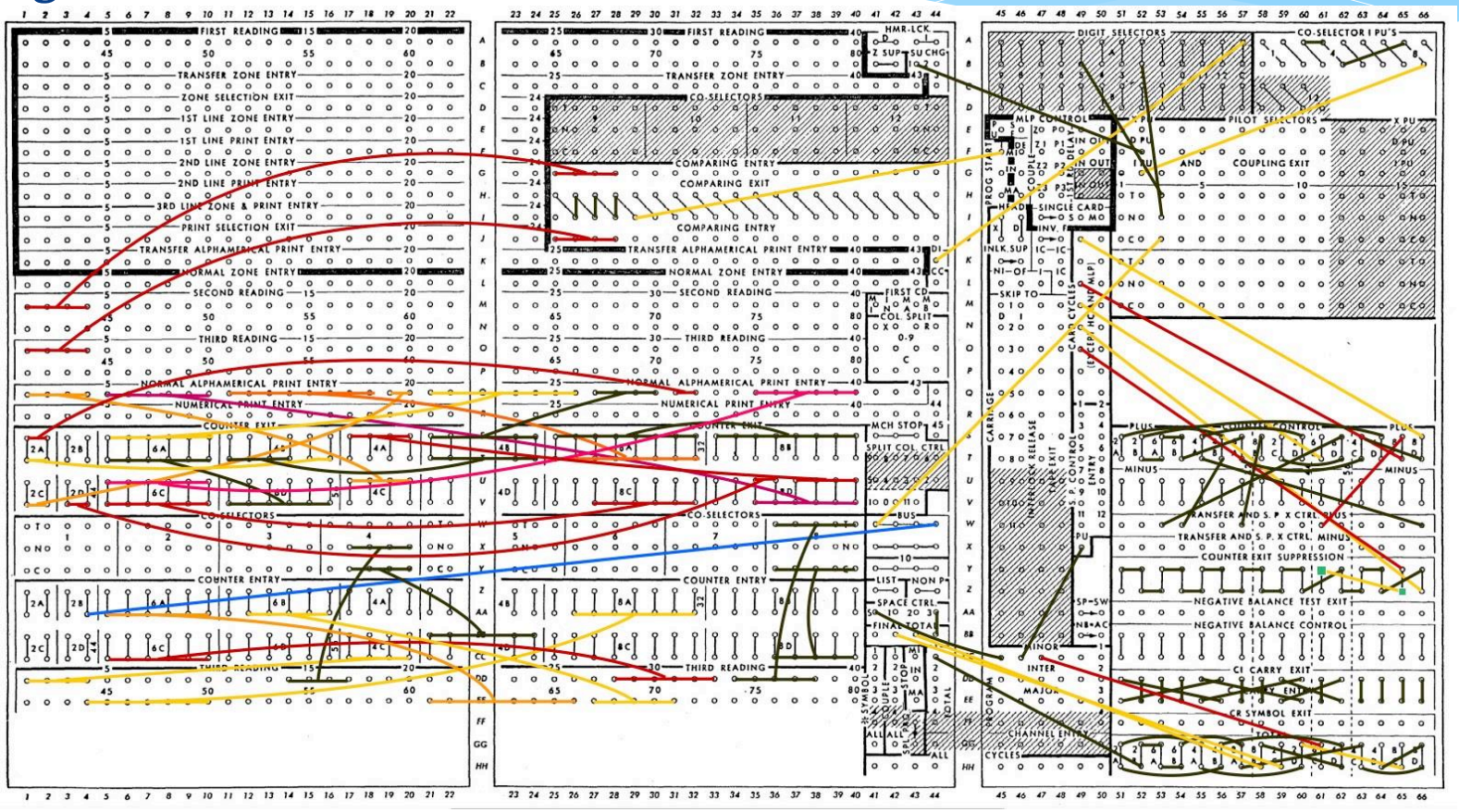
- 1946 ENIAC based on electric bulbs

- 18000 vacuum tube
- 70000 resistance
- 5 million metal connector
- *Faster but least reliable*



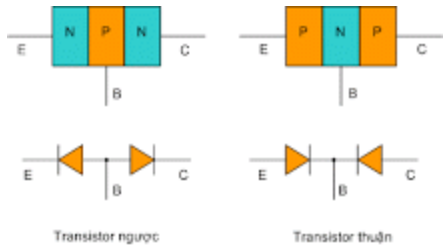
# Development history of electronic computers

- plug board

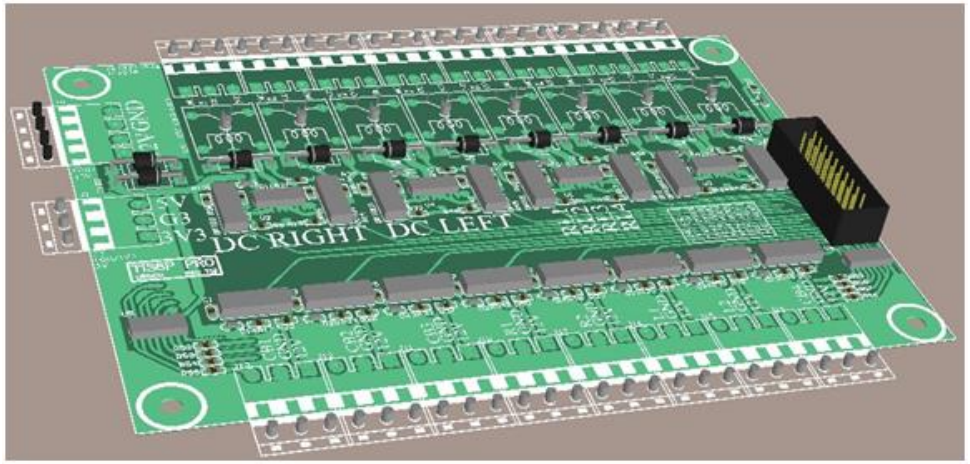


# Development history of electronic computers

- 1950-1958 Transistor
- 1959-1963 Semiconductor




- 1964-1974 Integrated Circuit (IC)



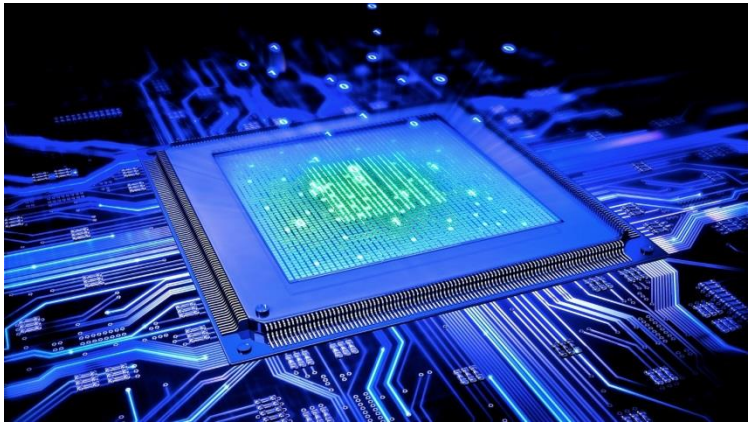
# Development history of electronic computers

- 1974-1990 Large scale IC:

Allow CPU, main memory or similar device to be produced in a single integrated circuit

-> new class of smaller, cheaper computer and parallel processor with multi CPUs

- 1990-now Very large scale IC, smart IC



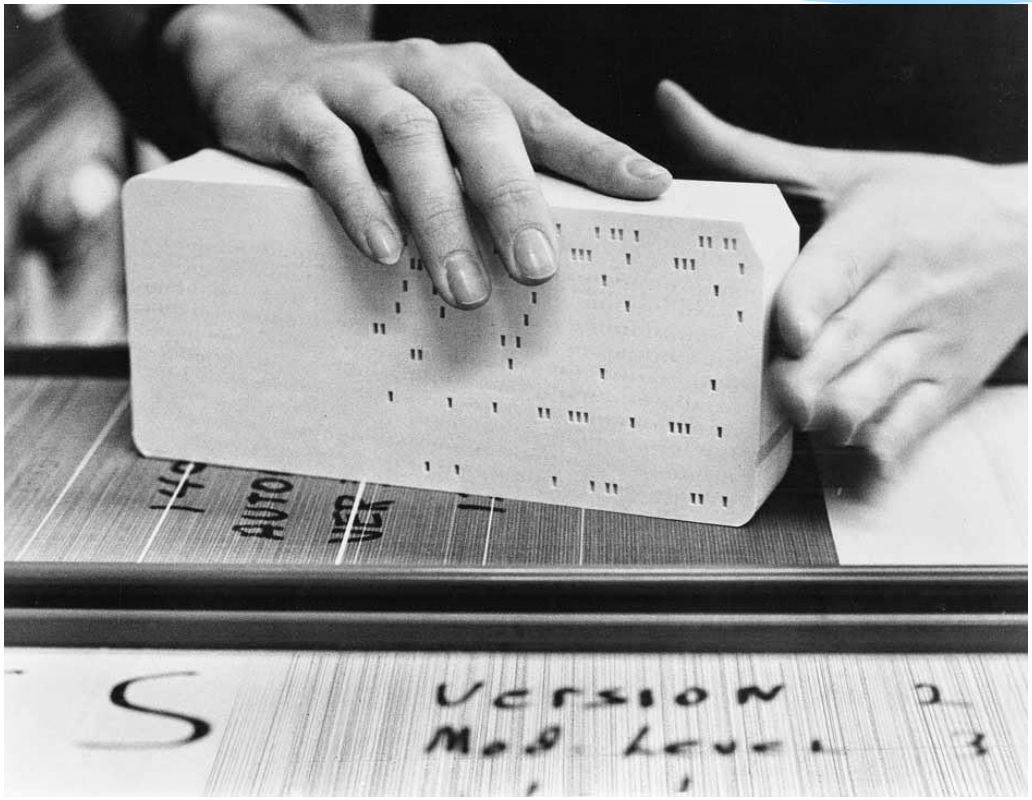
# Operating systems development's history

- History of electronic computer
- Operating systems development's history

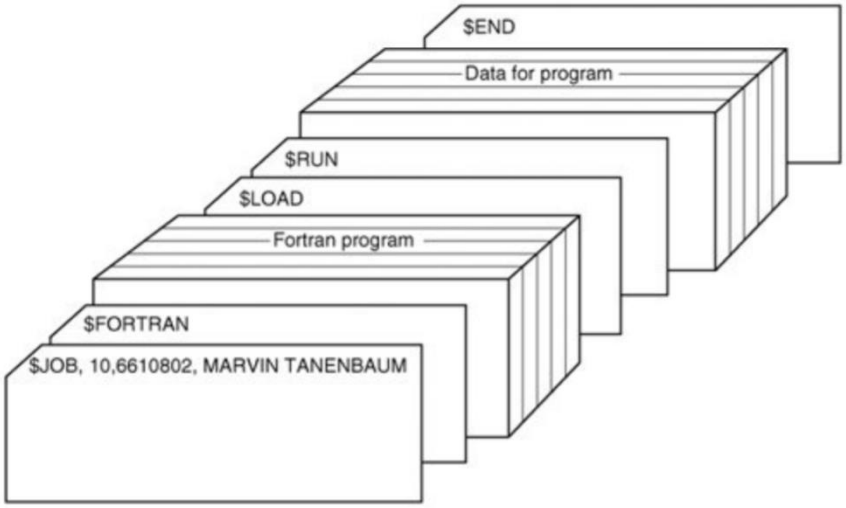
* 1948-1970 : Hardware expensive; human labor cheap
* 1970-1981 : Hardware cheap; human labor expensive
* 1981- : Hardware very cheap; human labor very expensive
* 1981- : Distributed system
* 1995- : Mobile devices

* 1948-1970 :
* Computer 1-5 M$ : Nation ‘s property, mainly used for military’s purposes ⇒ Require optimization for using hardware effectively
* Lack of human-machine interact
* User, programmer; operator are same group of people
* One user at a single time
  * User wrote program on punched cards
  * First card is bootstraps loader is loaded into memory and executed
  * Instructions in bootstraps loader fetch into memory and execute instructions on other later cards (application program)
  * Check light bulb for results, debug
* Debugging is difficult
* Waste processor time
* Solution: batch processing

* 1948-1970 :



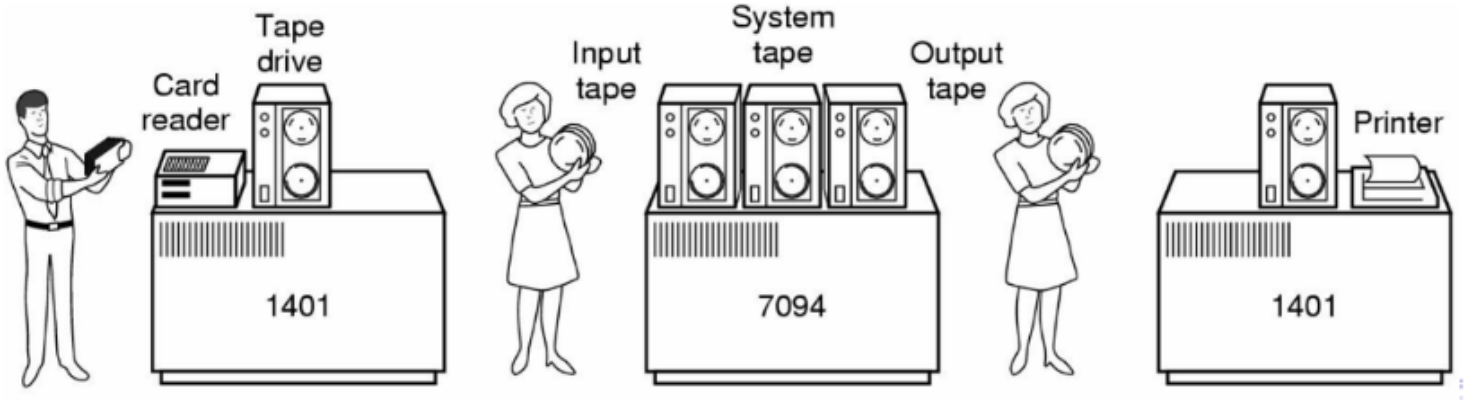
* Batch processing and professional operator
* Programmer give program to operator
* Operator group program into a single pack (batch)
* Computer read and run each program consequently
* Operator take the result, print out and give to programmer



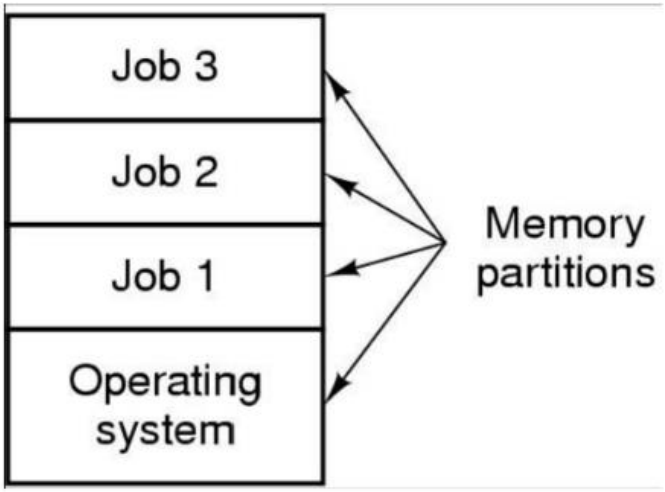

- Reduce waiting time between jobs
- Input/output problem
  - Computer getting faster
  - Card reader still slow

⇒CPU has to wait for card reading/writing

* Replace card reader by tape ⇒ Independent external computers for reading/writing data
* External devices are designed to be able to Direct Memory Access, using interrupt and i/o channel
  * OS request I/O device then continue its work
  * OS receive interrupt signal when I/O devices finishes
* ⇒Allow overlap between computing and I/O



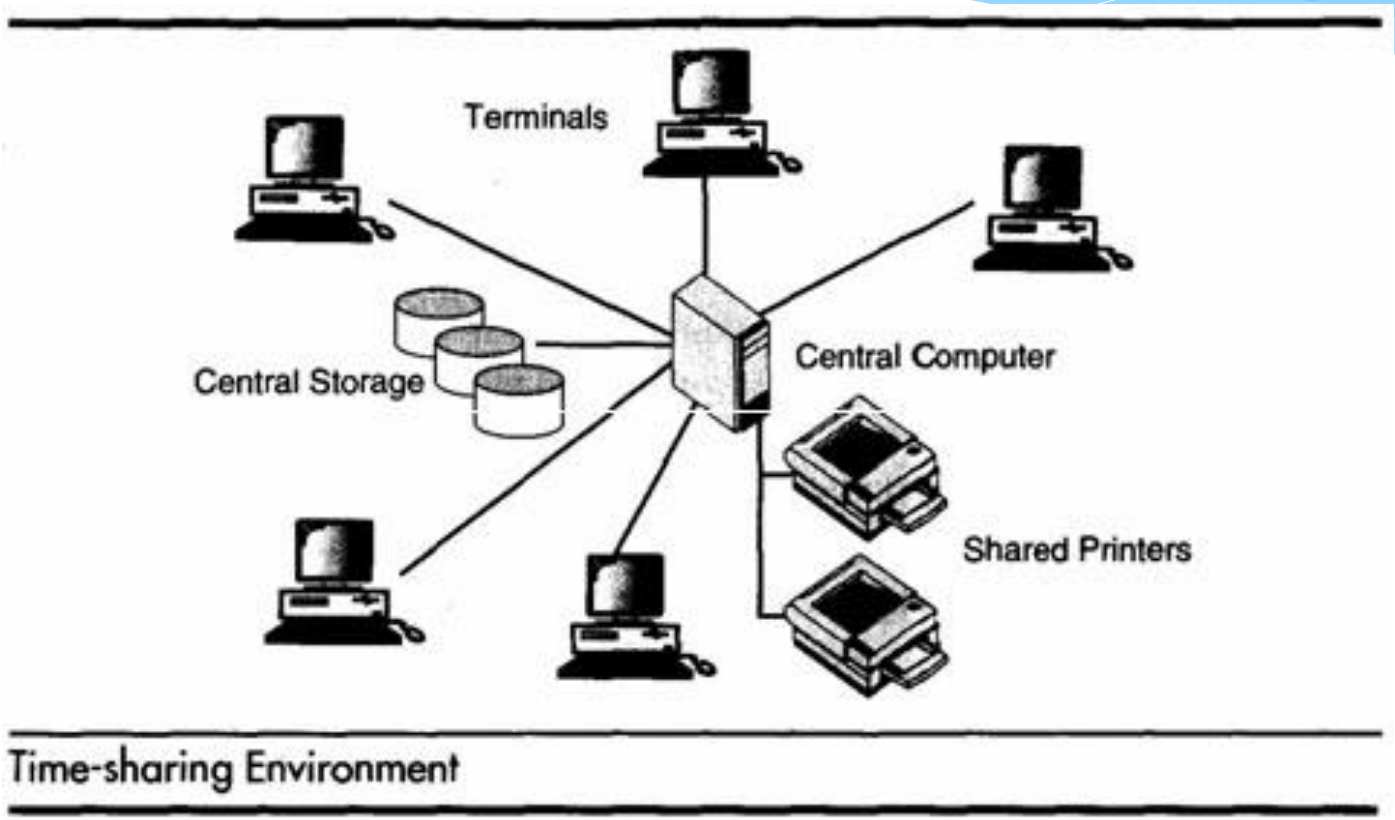
- CPU is reprogrammed to be switch easily between programs
- Hard ware: memory space larger and cheaper Some program can run simultaneously -> multi programming




  - More overlap between computing and I/O
  - Require memory protection between programs and keep one crashed program from damaging the system
- Problem: OS has to manage all interaction ⇒ out of control (OS360: 1000 error)

* 1970-1981 :
- Computers prices about 10.000$ ⇒ used widely for different jobs
- OS technology became stable.
- Using cheap terminal device (1000$) allow many user to interact with the system at the same time
  - User perform different works (text editor, chat, program debugging,..) ⇒ require system to be exploited effectively
    Example: a PC: 10M calculation/s; typing speed 0.2s/1 character => lost 2M calculation per one typing
  - ⇒ Time sharing operating system
    - Problem: system's response time
- Computer network was born (ARPANet : 1968) Communication between computer; Protection against network attack

# Time sharing system




Time-sharing Environment

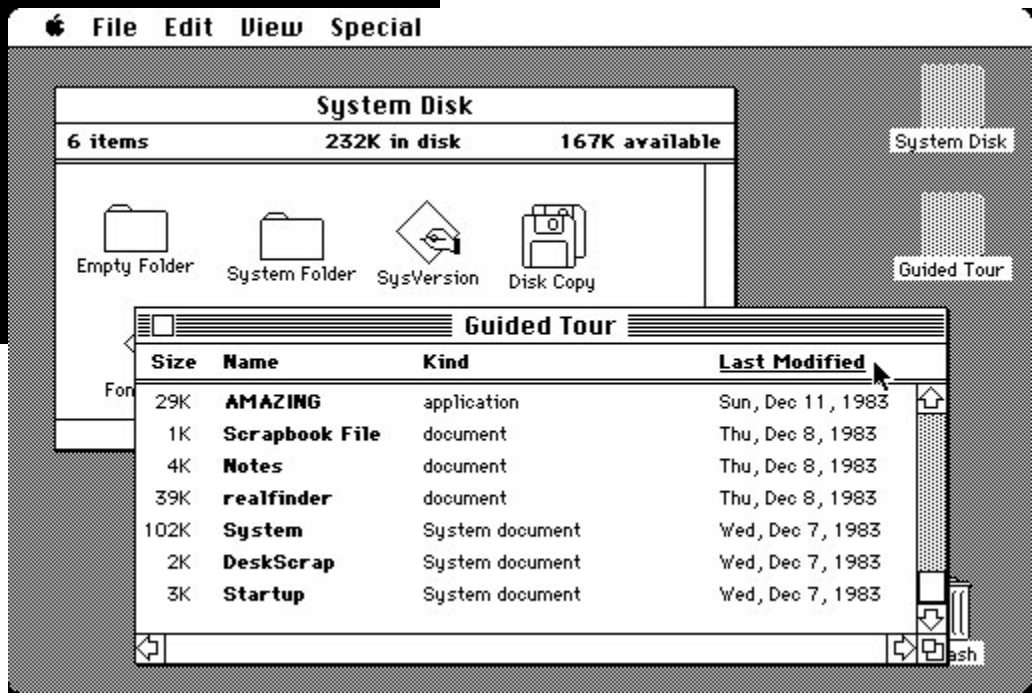
[FAIZAL MOHAMAD – Introduction to operating system]

* 1981-1995 :
- Computers prices about 1000$; human labor 100K $/year ⇒ Computer are used more widely for working more effectively
- Personal computing
  - Cheap computer, single person can afford (PC).
  - OS on PC
- Hardware resources are limited (Early : 1980s)
  - OS become library of available procedures
  - Run One program at a time (DOS)
- PC become more powerful
  - OS meet complex problems: multi tasking, memory protection… (WINXP)
- Graphical user interface (MAC, WIN,..)

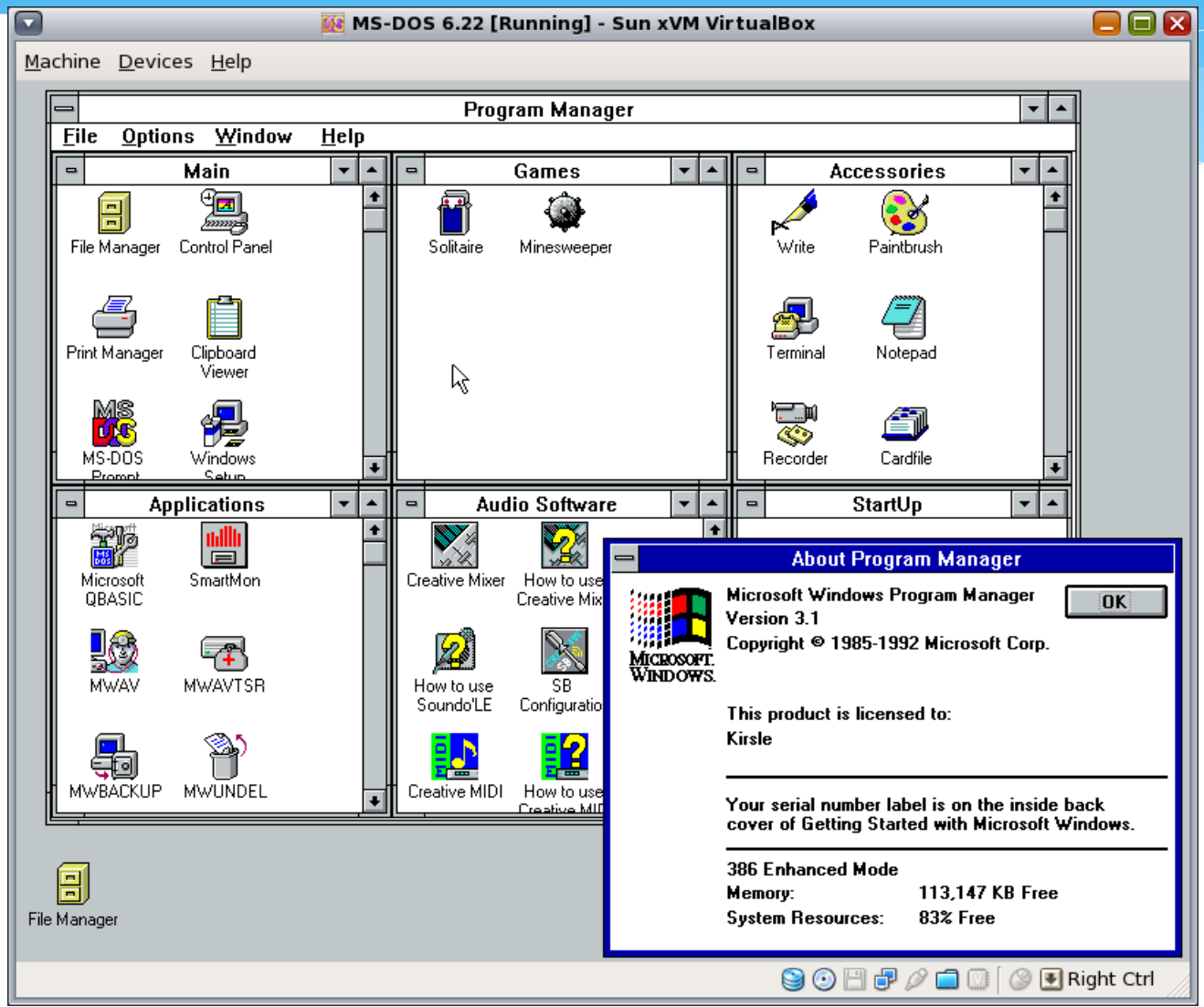
* DOS User interface

```
Welcome to FreeDOS

CuteMouse v1.9.1 alpha 1 [FreeDOS]
Installed at PS/2 port
C:\>ver

FreeCom version 0.82 pl 3 XMS_Swap [Dec 10 2003 06:49:21]

C:\>dir
 Volume in drive C is FREEDOS_C95
 Volume Serial Number is 0E4F-19EB
 Directory of C:\

FDOS                <DIR>  08-26-04   6:23p
AUTOEXEC BAT          435  08-26-04   6:24p
BOOTSECT BIN          512  08-26-04   6:23p
COMMAND  COM       93,963  08-26-04   6:24p
CONFIG   SYS          801  08-26-04   6:24p
FDOSBOOT BIN          512  08-26-04   6:24p
KERNEL   SYS       45,815  04-17-04   9:19p
        6 file(s)        142,038 bytes
        1 dir(s)   1,064,517,632 bytes free

C:\>_
```

* Macintosh User interface

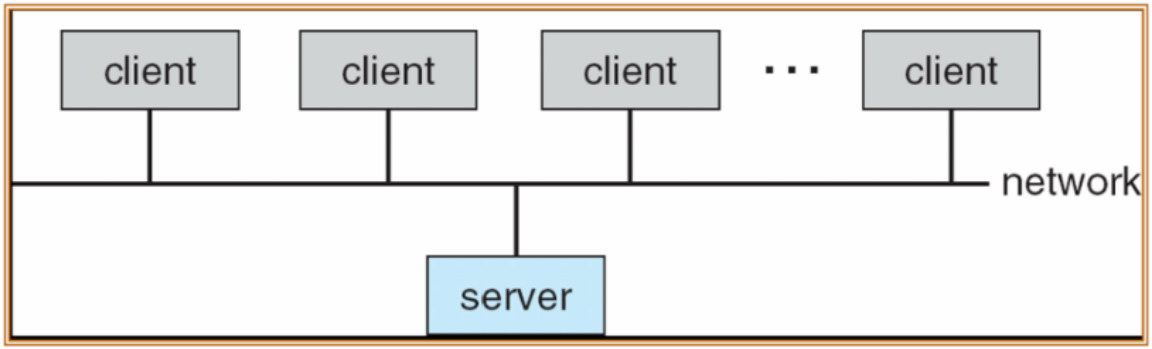


# Chapter 1. Operating System Overview
## 2.History of Operating Systems
### 2.2. History of Operating Systems



# Distributed systems

- Development time of networking and distributed operating systems




- Local area network
  - Computers share resources: printer, File servers,..
  - Client / Server model
- Services
  - Computing, storage
  - Services provided through Internet.
- Problems
  - Transmission delay; bandwidth, reliable...
  - Virus (love letter virus 05/2000),..

- >45 millions computers were infected
- Stole information
- Auto send emails from contact list
- Download Trojan

* Mobile devices
* Mobile devices become more popular
  * Phone, Laptop, PDA . . .
  * Small, changeable and cheap → More computers/human
  * Limited ability: speed, memory,...
* Wide area network, wireless network
  * Traditional computer divided into many components (wireless keyboards, mouse, remoting storage)
* peer-to-peer system
  * Devices with the same role working together
  * “Operating system’s” components are spread globally
* Cloud computing
  * Cloud operating system

* Conclusion

- The development of the operating systems are strongly connected with the computers' development

- Operating system development pulled the development of computers

# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

# Operating System definition and classification

- Definitions
- Classifications

## Observer's perspective

- Different objects have different requirements for operating system
- Different observing perspectives ⇒ different definitions

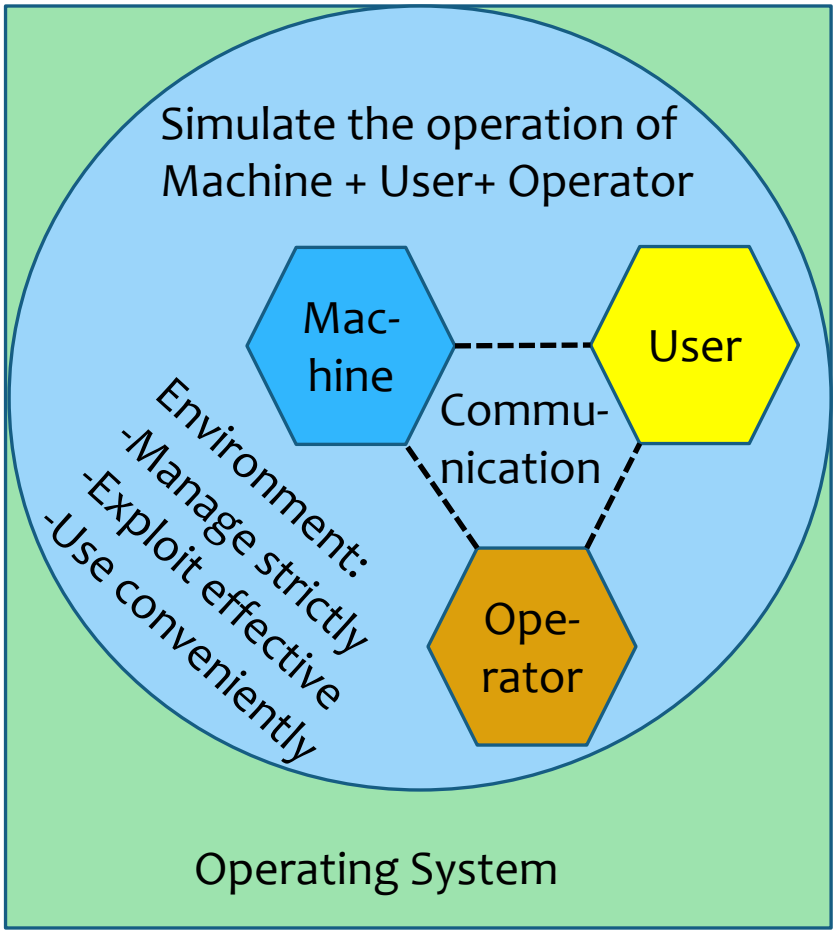
User: A system of programs that help exploiting the computing system conveniently

Manager: A system of programs that help managing computing system's resource effectively

Technical perspective: A system of programs equipped for a specific computer to make a new logic computer with new resource and new ability

System engineer perspective: A system of programs that modelize, simulate the operation of computer, user and operators. It work in a communicating mode in order to make a convenient environment for exploiting the computer system and maximum resource management.

# System engineer's perspective




Simulate 3 roles ⇒ require 3 types of languages

- Machine language
  - The only working language of the system
  - All other languages have to be translated into machine language
- System operation's language
  - OS commands (DOS: Dir, Del..; Unix: ls, rm,..)
  - Translated by the Shell
- Algorithm language
  - Programming language
  - Compiler

# Operating System definition and classification

- Definitions
- Classifications

- Batch processing single program system
- Batch processing multi program system
- Time sharing system
- Parallel system
- Distributed system
- Real-time processing system

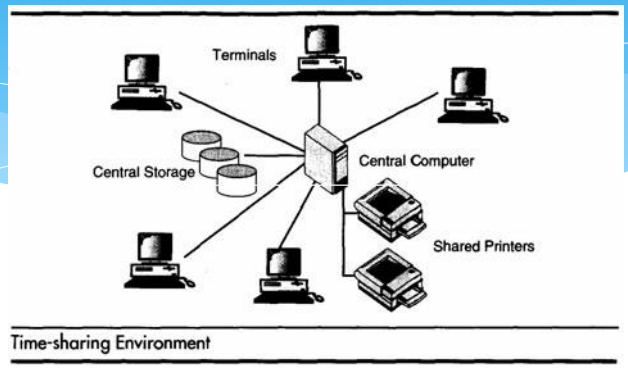
# Batch processing single program system

- Programs are performed consequently follow predetermined instructions
- When a program finished, the system auto run the next program without any external intervention
- Require a supervisor process the sequence of jobs and the supervisor has to stay permanent in the memory
- Need to organize a job queue
- Problem: when a program access an I/O device, processor has to wait

# Batch processing multi program system

- Allow many programs to run at the same time
  - Load one part of code and data of the programs into the memory (the remaining parts will be loaded at proper moments). The program are ready to run
  - Run the program similar to single program system
  - If the current program perform an IO, processor will be change to another program
- Save memory (no need to load all the program into the memory)
- Reduce processor spare-time
- High cost for processor scheduling. Which program can use processor next?
- Solve the memory sharing problem between programs

# Time sharing system



Time-sharing Environment

- Processor's usage allowance time is shared among ready-to-run programs
- Similar to batch processing multi program system (only load part of the programs)
- Processor is issued mainly by the operating system ⇒ how ? ⇒ Chapter 2
- Swapping times between program are small -> programs seem to run parallel
- Usually called: Multi tasking operating system (Windows)

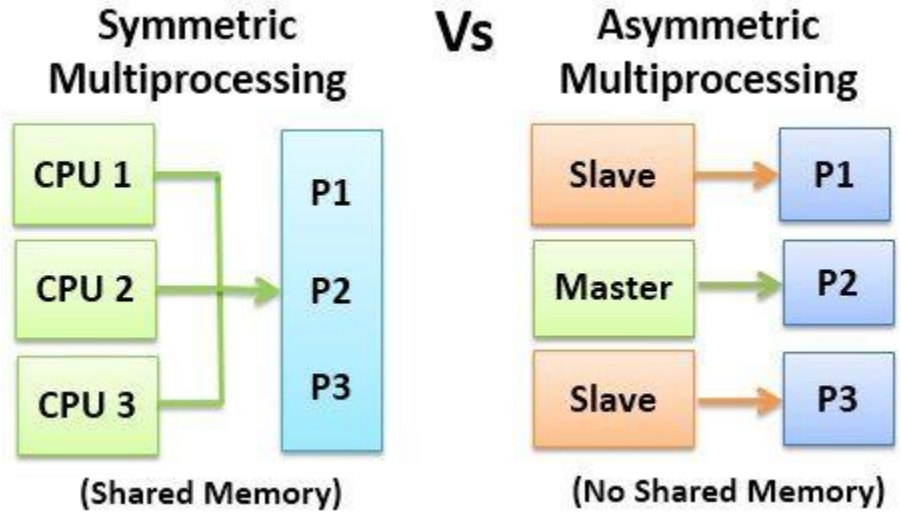
# Parallel system

- Constructed for system that has many processors
  - Many processors, works are done faster
  - More reliable: one processor breaks down will not affect the system
  - Advantage over single processor computer due to memory, peripheral devices sharing...

- Symmetric multi processing (SMP: symmetric)
  - Each processor run a single program
  - Processors communicate via a shared memory
  - Fault tolerance mechanism and optimal load balance
  - Problem: processor synchronization
  - Example: WinNT operating system

# Parallel system

- Asymmetric multi processing (ASMP: asymmetric)
  - One processor controls the whole system
  - Other processors follow the main processor's commands or predetermined instructions
  - This model has the master-slave relation form: The main process will make schedule for other processors
  - Example: IBM System/360



# Distributed system

- Each processor has a local memory and communicate via transmission lines
- Processors are different from sizes to functions (personal machine, workstation, mini computer,..)
- Distributed system is used for
  - Resource sharing：provide a mechanism for file sharing, remote printer…
  - Increase computing speed: One computing operation is divided into smaller parts and performed on different places at the same time.
  - Safety: One position get problem, others can continue working

# Real-time processing system

- Used mainly in controlling field.
- Solve a problem no late than a specific time.
  - Each problem has a deadline
  - The system must generate correct result in a determined time period
- This OS requires highly cooperate between software and hardware.

# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

# Basic Properties of Operating Systems

- High reliability
- Secure
- Effectiveness
- General overtime/ Inherit and adaption
- Convenience

## High reliability

- Every actions, notations have to be accurate
  - Only provide information when it's surely correct
    - When error happens: notify and stop the proceed or let the user decide
  - Require support from device
- Example: C:/>COPY C:/F.TXT A:

High reliability

- Example: C:/>COPY C:/F.TXT A:
  - Check the syntax of command copy
  - Check I/O card (motor, drive accessibility)
  - Check for file F.TXT existence in C drive
  - Check A drive
  - Check if file F.TXT already existed in A drive
  - Check if there is enough space in A
  - Check if the disk is write protection
  - Check written information (if required)
  - ....

# Basic Properties of Operating Systems

- High reliability
- Secure
- Effectiveness
- General overtime/ Inherit and adaption
- Convenience

## Security

- Data and programs have to be protected
  - No unwanted modification in every working mode
  - Secure from illegal access
- Different resources have different protection requirements
- Many levels protections with various of tools
- Important for multi tasking system

# Basic Properties of Operating Systems

- High reliability
- Secure
- Effectiveness
- General overtime/ Inherit and adaption
- Convenience

# Effectiveness

- Resources are exploited thoroughly;
- Resource that is limited still able to handle complex requirement.
- The system need to maintain the synchronization;
  - Slow devices do not affect the whole system operation

# Basic Properties of Operating Systems

- High reliability
- Secure
- Effectiveness
- Generalizable overtime/ Inherit and adaption
- Convenience

## Generalizable overtime

- System must be Inheritable
  - Operations, notification can not change
  - If changed: notify and with detailed guide (chkdsk/scandisk)
  - Help keeping and increasing users
- System must have ability to adapt to changes that may happen
- Example: Y2K problem; FAT 12/16/32

# Basic Properties of Operating Systems

- High reliability
- Secure
- Effectiveness
- General overtime/ Inherit and adaption
- Convenience

## Convenience

- Easy to use
- Various effective levels
- Have many assisting system

# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

# Notions in operating systems

- Process and Thread
- System's resources
- Shell
- System calls

## Process

- A running program
  - Codes: Program's executable instruction
  - Program's data
  - Stack, stack pointer, registers
  - Information that is necessary for running program
- Process >< program
  - Program: a passive object, contains computer's instructions to perform a specific task
  - Process: program's active state.

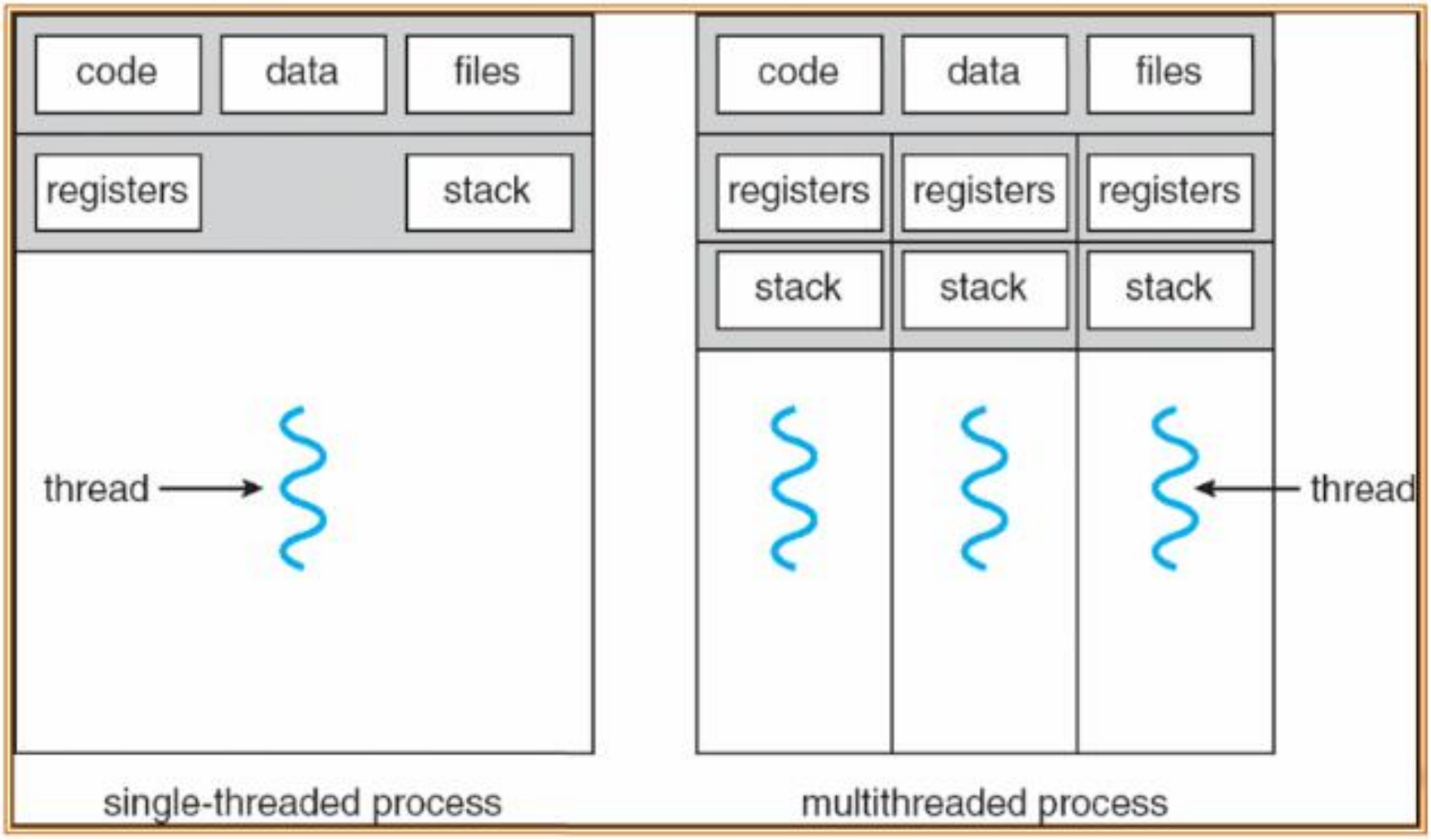
## Multi-process timesharing system:

- Periodically: OS pause one process and start another process
  - Need to store processes' information ⇒ process table
- One process can start other process
  - Ex: OS's Shell start a process to perform the command; when the command is done, terminate the started process
- Process can exchange information
- One process can include many threads

## Thread

- A sequence/thread of instructions executed in the program
  - Executable code, data
  - Instruction pointer, stack, registers
- Heavyweight Process: process contains one thread
- Lightweight process: contains more than one thread
- Multi_Threading model:
  - Threads running parallel, sharing process's global variables

# Chapter 1 Operating System Overview
## 5. Notions in operating systems
### 5.1. Process and Thread



# Notions in operating systems

- Process and Thread
- System's resources
- Shell
- System calls

Definition

- Everything that is necessary for a program to be performed
  - Space: System's storage space
  - Time: Instruction executing time, data accessing time
- System's resources
  - Memory
    - Distinguished by: Storage size, directly access, sequent access
    - Levelled by: main memory/internal; extend, external
    - Notions' distinguish: memory (physical area that contain data) and memory access (the process of searching for the data's location in memory)

# Chapter 1 Operating System Overview
## 5. Notions in operating systems
### 5.2 System's resources

- System's resources
  - Memory
  - Processor
    - System's most important component
    - Access level: instruction
    - Processing time
    - Multi-processor system: each processor's time is managed and scheduled independently
  - Peripheral devices
    - Retrieve, output information (I/O device)
    - Attached to the system via controller
    - Commonly considered peripheral devices-controller devices

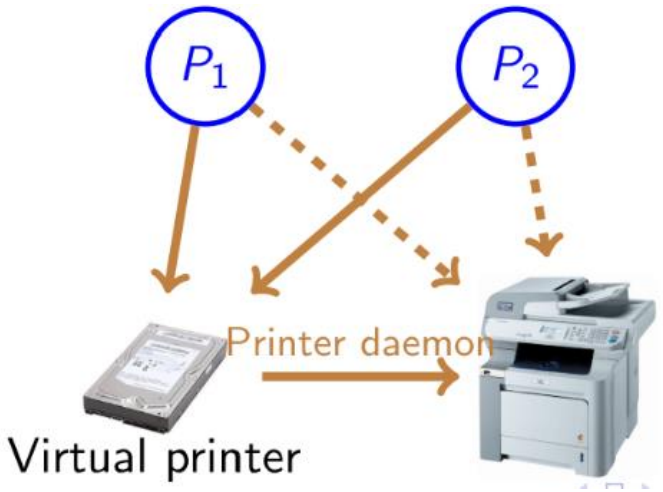
# Resource's classification

- Resource's types
  - Physical resource: physical devices
  - Logic resource: variable; virtual devices
- Sharing ability
  - Sharable resource: at a specific time, it can be allocate for different processes. Example: Memory
  - Non-sharable but dividable resource: Processes use the resource follow an order; Example: processor
  - Non-sharable and non-dividable resource: at a specific time, only one process can use the resource. Example: Printer

# Virtual printer

- Resource allocated for user's program in a changed form
- Only appear when the system requires or when the system creates it
- Automatically disappear then the system terminate or more precisely, when the process that works with this resource terminates

Example: Virtual printer



# Notions in operating systems

- Process and Thread
- System's resources
- Shell
- System calls

- A special process: user and OS communication environment
- Task
  - Receive user's command
  - Analysis received command
  - Generate new process to perform command's requirement
- Receives command from command's line or graphical interface
- Single task environment (MS-DOS):
  - Shell will wait until one process finishes and then receive new command
- In multi-tasking system (UNIX, WINXP,. . .) After creating and running new process, Shell can receive new command

# Chapter 1 Operating System Overview

## 5. Notions in operating systems

### 5.3 Shell

# Notions in operating systems

- Process and Thread
- System's resources
- Shell
- System calls

- Provides environment for interacting between user's program and the operating system
  - Programs utilize system calls to request services from operating system
    - Create, delete, use other software objects operated by the operating system
  - Every single system call is corresponding to a library of sub-programs (functions)
- System calls are done in the form of
  - Instructions in low-levels programming languages
  - Interrupt requests (Int) in assembly language
  - API functions calls in Windows
- Input parameters for the services and returned results are located in special memory areas
  - For example: when making request for an interrupt, the function name is stored in the register AH
  - Int 05 : print to monitor ; Int 13/AH=03h : DISK – WRITE/ DISK SECTOR

## Example

| Func BOOL WINAPI ExitWindowsEx(int uFlags, int dwReason); | |
|---|---|
| **uFlags** | **Shutdown types** |
| EWX_LOGOFF<br>EWX_POWEROFF<br>EWX_ REBOOT | End process and exit Windows<br>Shutdown system and turn off computer<br>Shutdown and restart computer |
| **dwReason** | **Reason for shutdown** |

**File log_off.c**

```
#include␣<windows.h>
int␣main(int␣argc,␣char␣*argv[]){
␣␣␣␣ExitWindowsEx(EWX_LOGOFF,␣0);
␣␣␣␣return␣0;
}
```

# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

# Chapter 1 Operating System Overview

## 6. Operating System structures

- System's components
- Operating system's services
- System calls
- Operating system's structures

- Process management
- Main memory management
- Input Output system management
- Files management
- Storage memory management
- Data transmission system (network)
- Protection system
- User interface

## Process management

- Process: A running program
- Process utilize system's resources to complete its task
  - Resources are allocated when process created or while it's running
  - Process terminates, resources are returned
- It is possible for many processes to exist in the system at the same time
  - System process
  - User process
- The tasks of OS in process management
  - Create and terminate user's process and system's process
  - Block or re-execute a process
  - Provide mechanism for process synchronization
  - Provide method for processes' communication
  - Provide mechanism for controlling deadlock among processes

## Main memory management

- Main memory: an array of byte(word); Each element has an address; where data are accessed by CPU
- To be executed, a program must be given an absolute address and loaded into main memory. When the program is running, the system access instructions and data in main memory.
- To optimize CPU time and computer's speed, some processes are kept in memory
- The role of OS in main memory management
  - Store information about used areas in memory and who used them
  - Decide which process will be fetched into main memory when the memory is available.
  - Allocate and retrieve memory when it's necessary

## Input-Output system management

- Objective: hide physical devices' details from users to help them operate easier.

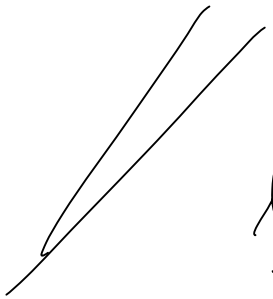
- Input-Output system management includes
  - Memory management of buffering, caching, spooling
  - Communicate with device drivers.
  - Controller for special hardware devices. Only device driver understand its associated-device's specific structure

## File management

- Computer can store information on many types of storage devices
- File: storage unit
- File management task
  - Creates/ deletes a file/directory
  - Provides operations over files and directory
  - Reflects file on secondary storage system
  - Backs up file system on storage devices

## Storage memory management

- Program is stored in secondary memory (magnetic disk) until it's fetched into main memory and executed.
- Disk is utilized for storing data and processed result.
- Data and result can be stored temporarily on disk: virtual memory
- The role of operating system in disk management
  - Unused area management
  - Provide storage area as requested
  - Schedule disk accessing effectively



Some mechanism will later be learnt and included in final exam

## Data transmission system (Distributed system)

- Distributed system combined of set of processor (sym/asym) without common clock and memory. Each processor has a local memory.
- Processor connected via transmission network
- Transmission is performed via protocols (FTP, HTTP...)
- Distributed system allow user to access different resource
- Access to sharing resources will allow
  - Increase computing speed
  - Increase data availability
  - Increase the system reliable

## System's protection

- Multi users operate with the system at the same time ⇒ Processes have to be protected from other processes' activities
- Protection is a controlling mechanism of program or user's access to system or resource
- Protection mechanism will require
  - Distinguish between legal or illegal usage
  - Set imposed controls
  - Provide tools for imposing

## User interface

- Perform user's command. Commands are provide for operating system 's command controller to
  - Create and manage process
  - Manage main memory and storage memory
  - Access file system
  - Protect
  - Network system
  - . . .
- User interface can be command line (DOS, UNIX) or more friendly with graphical interface (Windows, MacOS)

## User interface

Some forms of human-computer interact

- Command line
  - Simple but organized
  - Do not require complex system specification
  - Easy to add parameter
- Selection table
  - Menu
  - Popup
  - Menu_popup: 2 method: on and onselect
- Symbol

  window, icon, desktop

# Chapter 1 Operating System Overview

## 6. Operating System structures

- System's components
- Operating system's services
- System calls
- Operating system's structures

## Main and basic services

- **Program execution**: the system is able to load the program into memory and execute it. Program must be finish execution in a normal or abnormal (has error) way.
- **Input-output operations**: To increase the performance, programs do not directly access IO devices. The OS has to provide means to perform I/O.
- **File system operations: Program is able to read, write, create or delete file.**
- Communication: Information exchange between running process on the same computer or different computer in the network.
- **Communication is performed via sharing memory or message transferring technique.**
- **Error detection: Confirm work correctly by showing an error is from** CPU, memory, in devices or in programs. Each type of error, OS has a corresponding way to handle.

## Support services

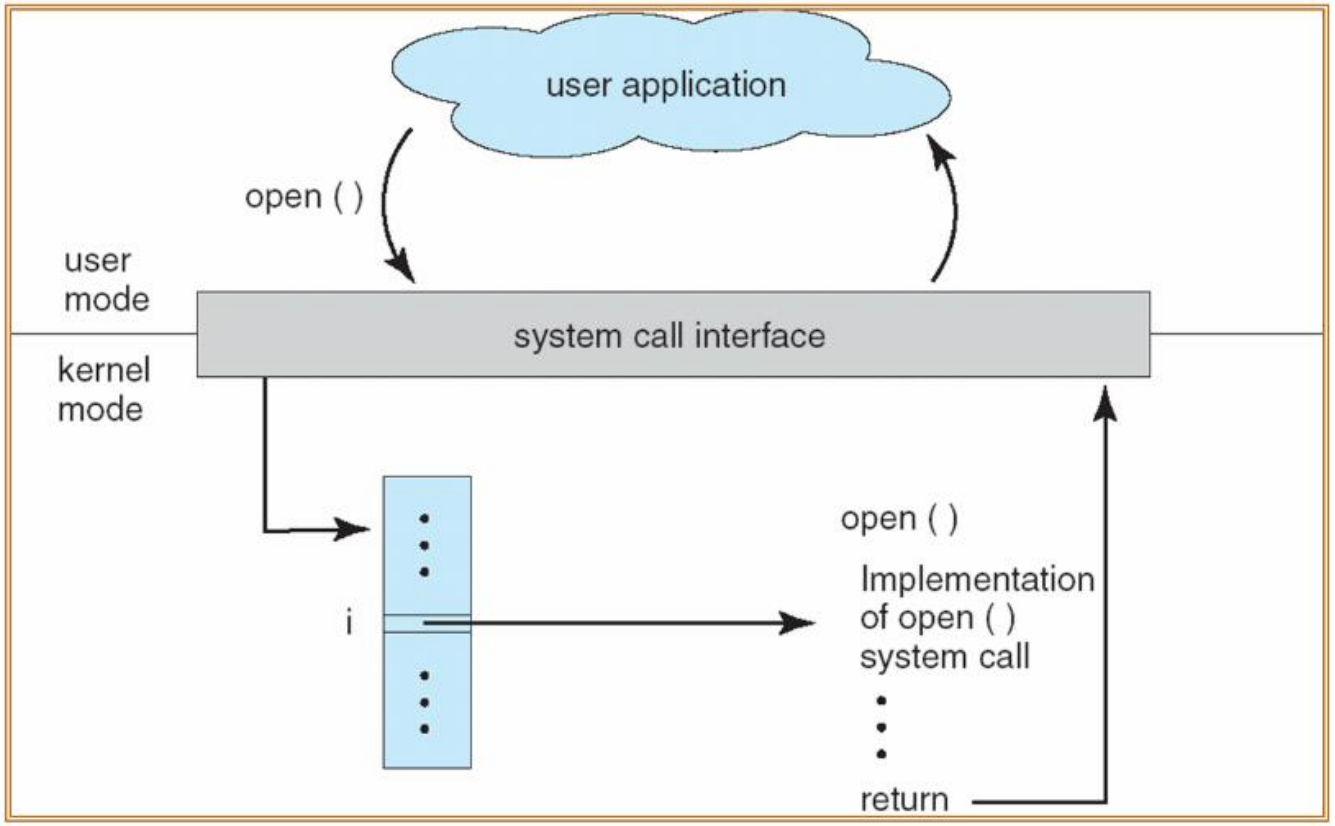
Not for user but for operate the system effectively

- **Provide resources** Allocate resource for many users or tasks to perform at the same time
- **Report statistics** Store information of types and amount of used resources for computing (usage cost), research (system improvement)
- **Protection** Ensure all access to system resources are controlled

- System's components
- Operating system's services
- System calls
- Operating system's structures

# System call

Provide an interface between process and operating system



## System call

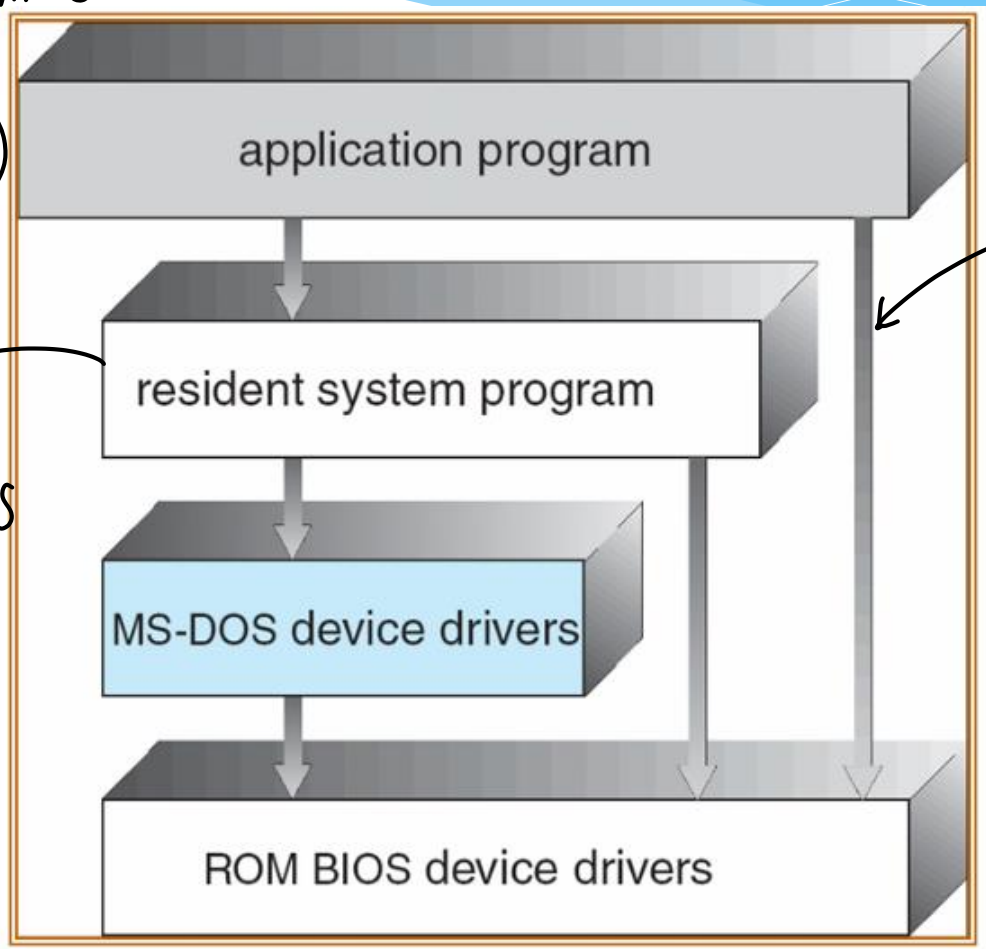
- Process management: initialize, terminate process..
- Memory management: allocate and free memory…
- File management: create, delete, read and write file…
- Input Output device management: perform input/output…
- Exchange information with the system. Example get/set time/date…
- Inter process communication

- System's components
- Operating system's services
- System calls
- Operating system's structures

# MS-DOS structure (Silberschatz 2002)

(single-program OS

(can run at most 1 program at a time)



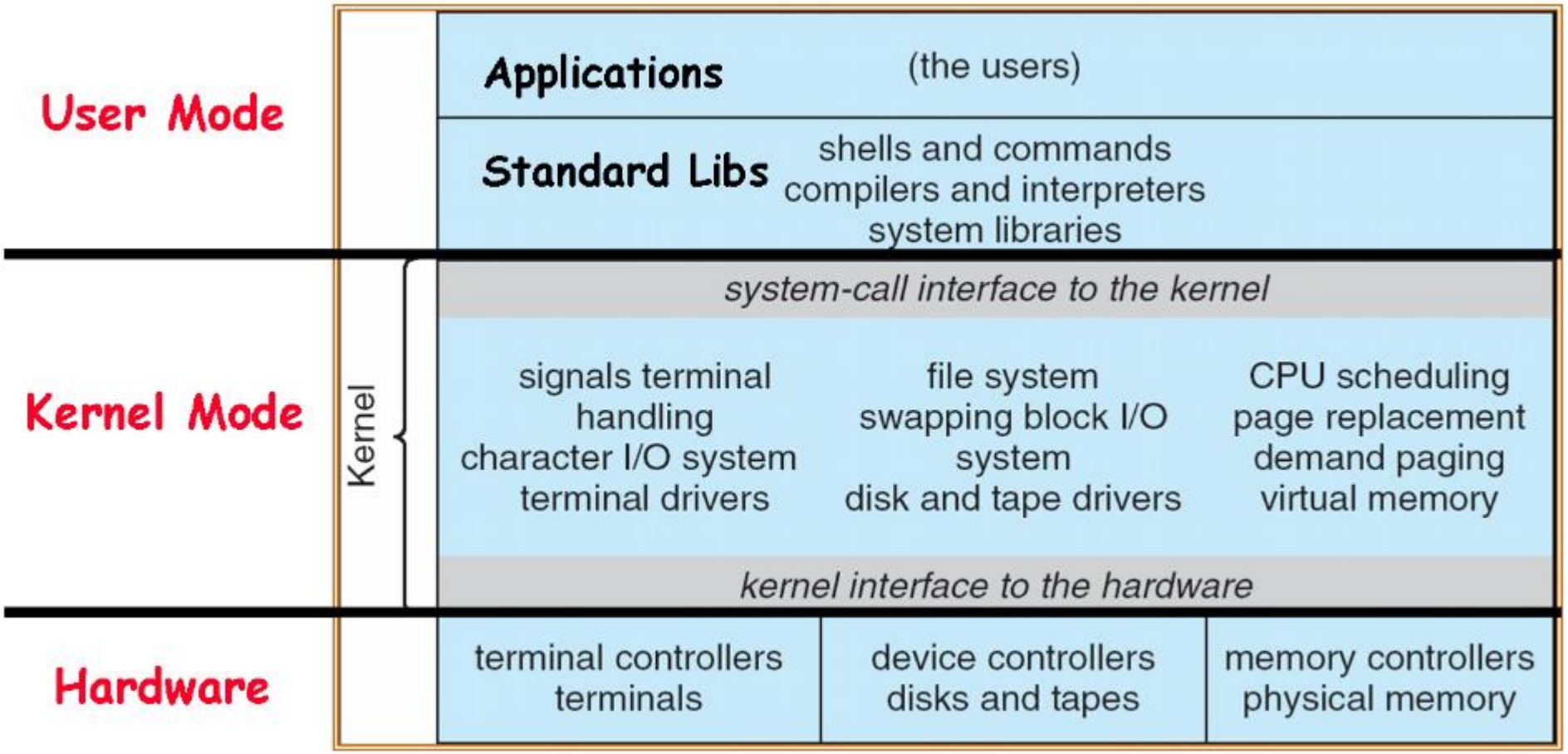

programs that stay permanent inside the OS

even the user can access the lowest lvl of sys

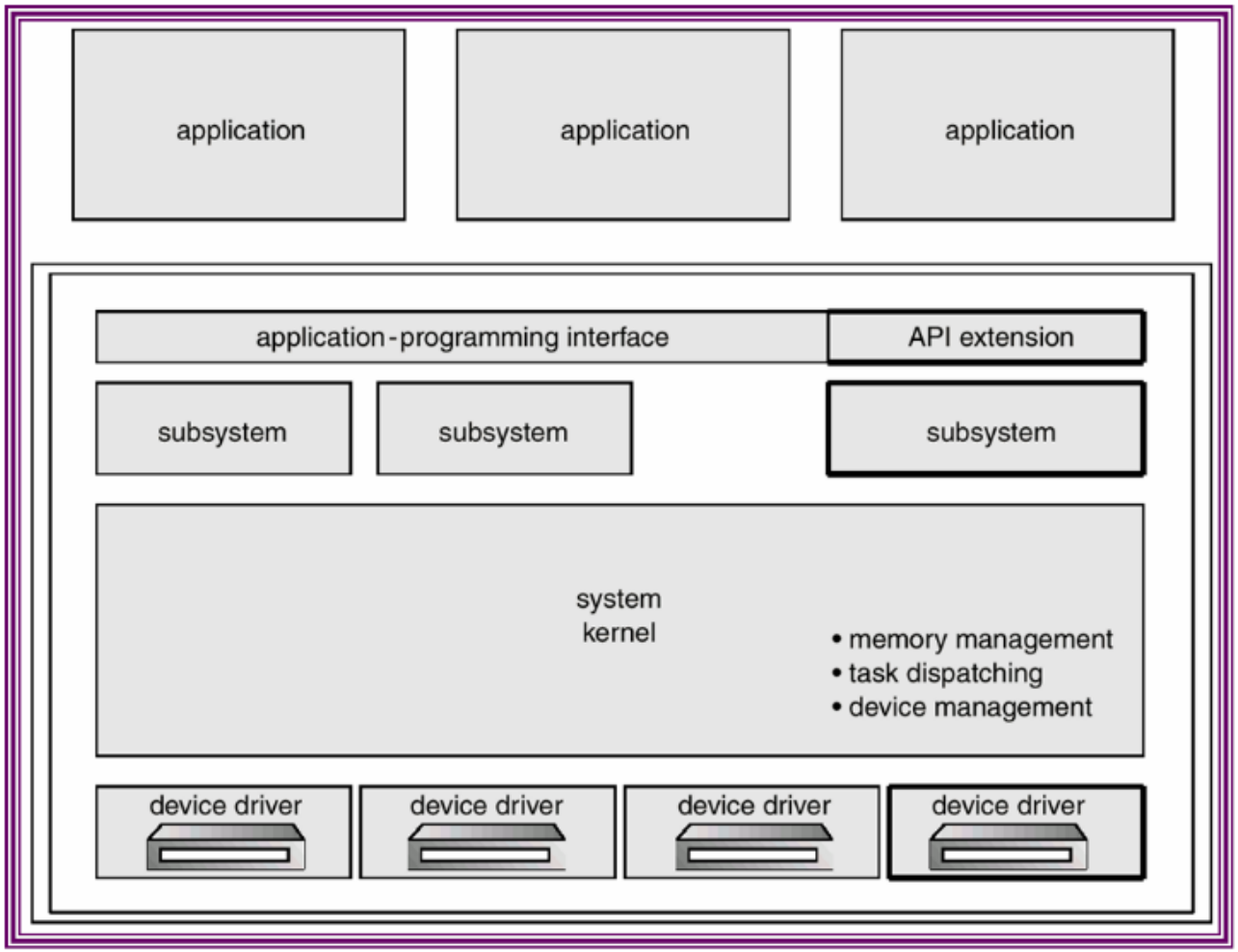
↓

no protection

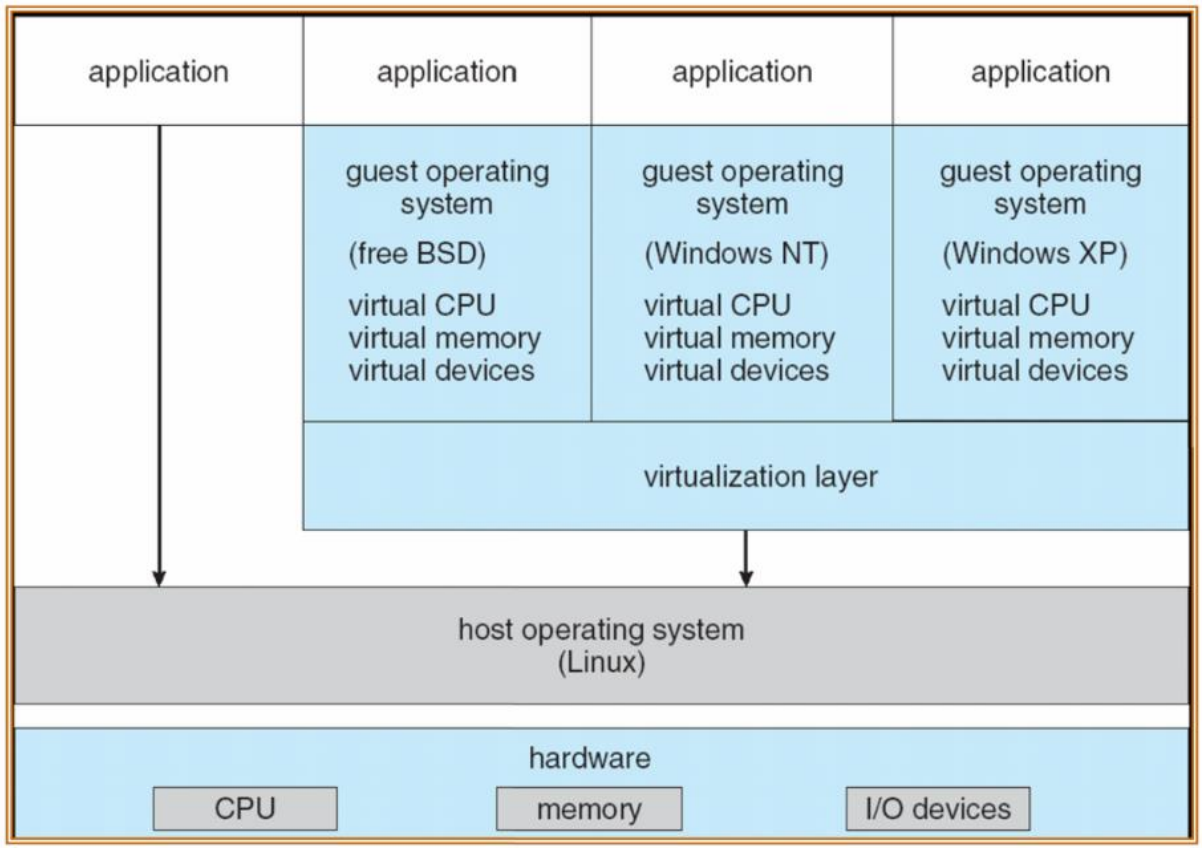
# UNIX structure (Silberschatz 2002)



# OS/2 structure (Silberschatz 2002)

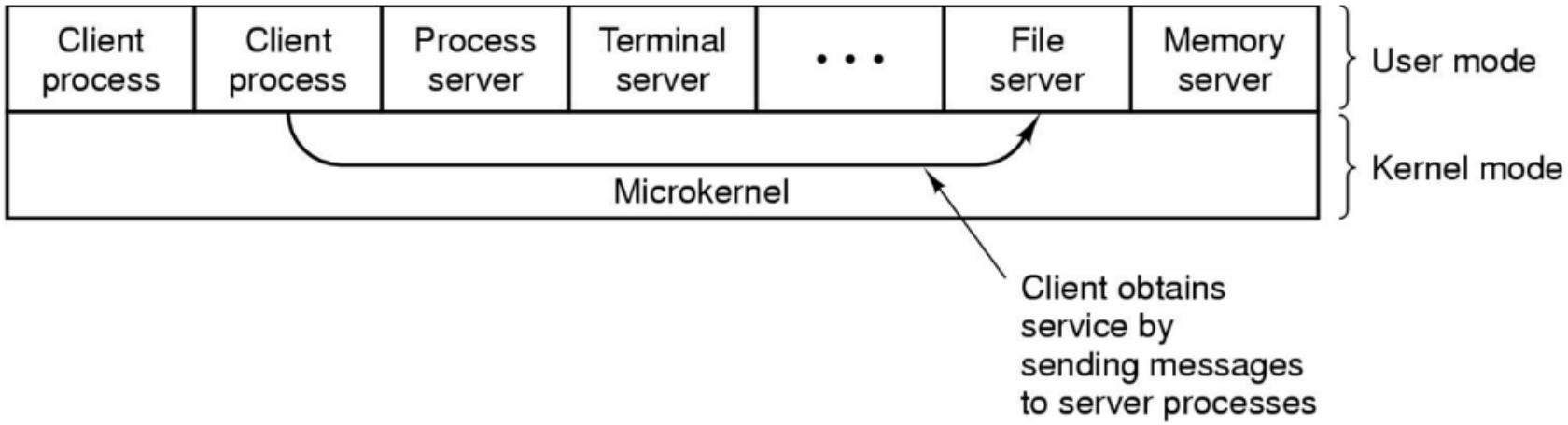


# Virtual machine(Silberschatz 2002)



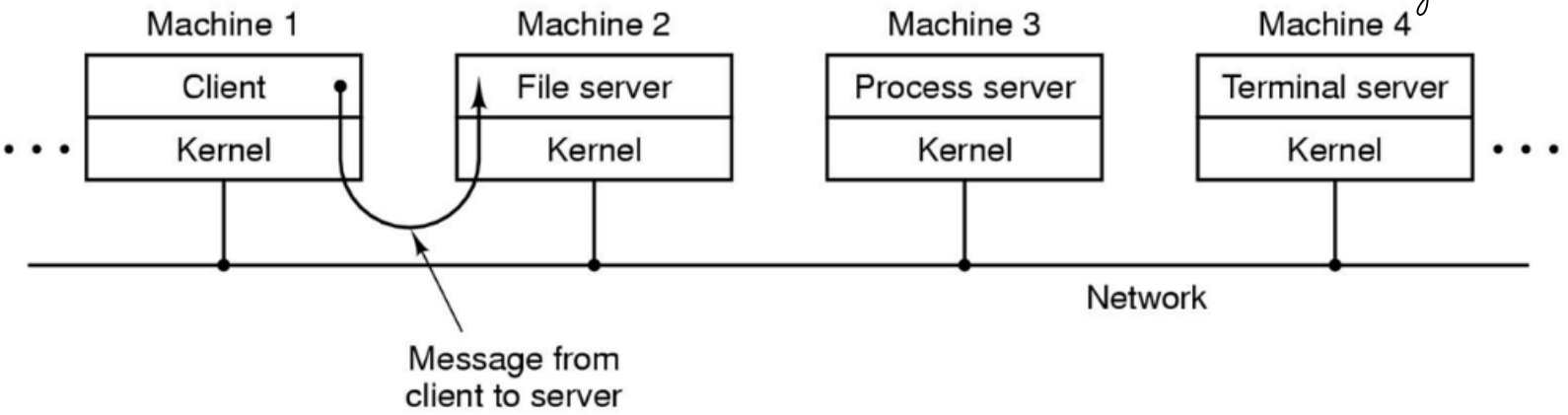

VMWare architecture

# Client-Server Model (Tanenbaum 2001)



# Client-Server model in distributed OS (Tanenbaum 2001)

using micro-kernel
↳ OS components are made portable; not part of the kernel
⇒ OS can be distributed, kernel for communication only




→ Modern OS uses a hybrid approach

Monolithic VS. Micro-kernel

# Chapter 1 Operating System Overview

① Operating system notion
② History of Operating Systems
③ Definition and Classifications
④ Basic Properties of Operating Systems
⑤ Notions in operating systems
⑥ Operating System structures
⑦ Principles of Operating Systems

# 5 Basic Properties of OS

① High-reliability : any notification, in4 is accurate
- ↑ hardware support
- ↳ by checking

② Security : user + program should be protected ; ∄ unwanted access/modification

③ Effectiveness : sys should exploit the resources
- ↳ handle complex requirements on limited resources
- ↳ protect sys from 1/some slow devices

④ Generalization over time (Inherit / Adaptability)
- ↳ Adapt to changes/incident that may happen
- ↳ Inherit to serve more users

⑤ Convenience

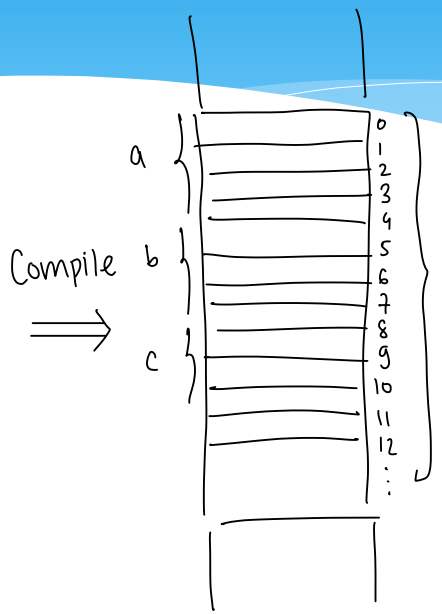
Can also applied for softwares

# Principles of Operating Systems

- Modul principles
- Relatively localization principle
- Macroprocessor principle
- Initialization when start-up principle (khởi tạo ⊙ cài đặt)
- Function overlap principle
- Standard value principle
- Multi-level protections principle

# Relative localization principle

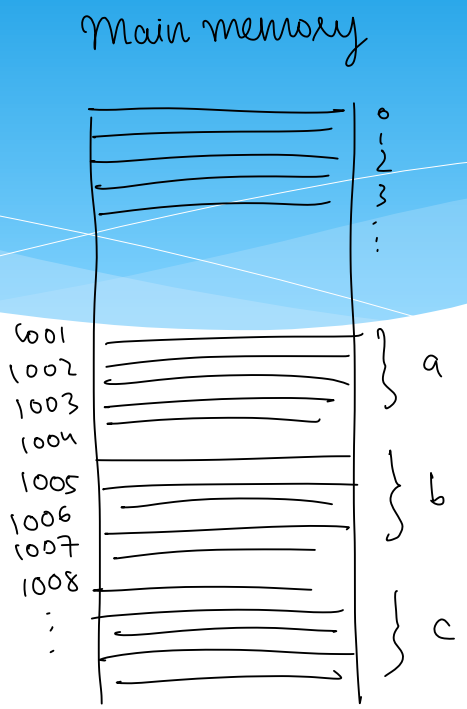
```
int a;
int b;
float c;
void main() {
   c = a+b;
   ...
}
```




Compile ⟹

The program relative address (addr of variable is the addr relative to the program's)

Execute ⟹ LOAD




Everytime we execute a program, it will be loaded onto another area in main memory!

↳ that's why we can NOT use absolute address in our code

↳ We use the relative address

(Các câu lệnh ở đây sử dụng địa chỉ tương đối)

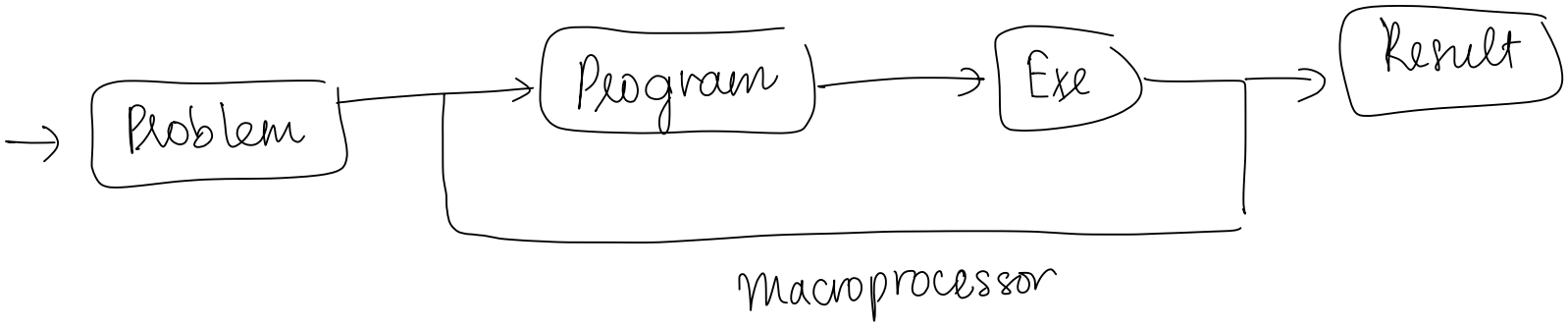
# Macroprocessor Principle

(các lệnh trên máy cta gọi là các 'macro')

① Receive requests
② List out modules that can satisfy user's requests
③ Merge these modules onto 1
④ Run this new process ⇒ Return

※ OS works like a macro processor, not a single processor (do the task itself)

(does not prep the prog to serve users beforehand)



# Initialization when Start-up / Generate principle

OS is made up of modules in the available module repo to satisfy

- machine's config
- user's request

Set-up

Select module / Set parameters / Choose dest folder

↓

Install

# Principle of Functional Repetition

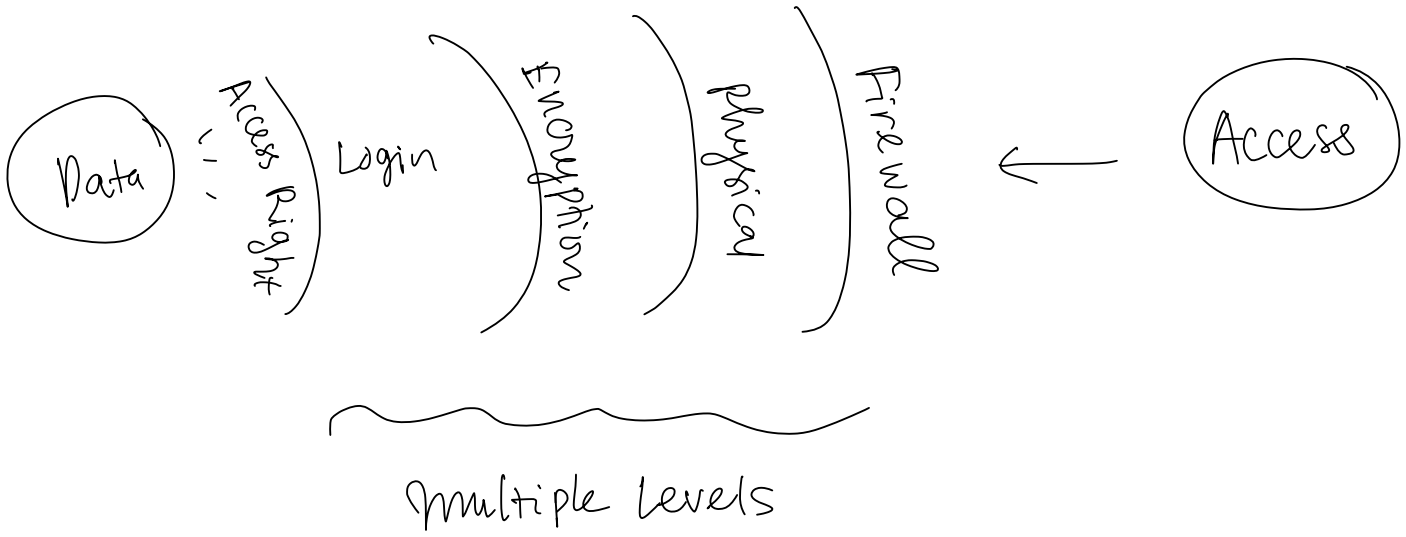
# Standard Value Principle

- Module parameterization → ↑ module flexibility
- (−) Need to remember parameters (nb, meaning of each...) so that we do NOT give the wrong inputs
  - ↳ solution: default/standard values (kept in some special tables)
    - ↳ for convenience + safety
    - special tables → Windows registry

Different modules / parameters → serve → Different tasks

# Multi-level protection

Different resources → Diff. characteristics → Diff. methods of protection



# Chapter 1 Operating System Overview

## Summary

① Notion of Operating system
- Layering structure of OS
- OS's functions

② History of Operating system
- History of computers
- History of Operating system

③ Definition and classification of OS
- Definitions
- Classification

④ Basic properties of OS
- High reliability
- Security
- Effective
- Generalize overtime
- Convenience

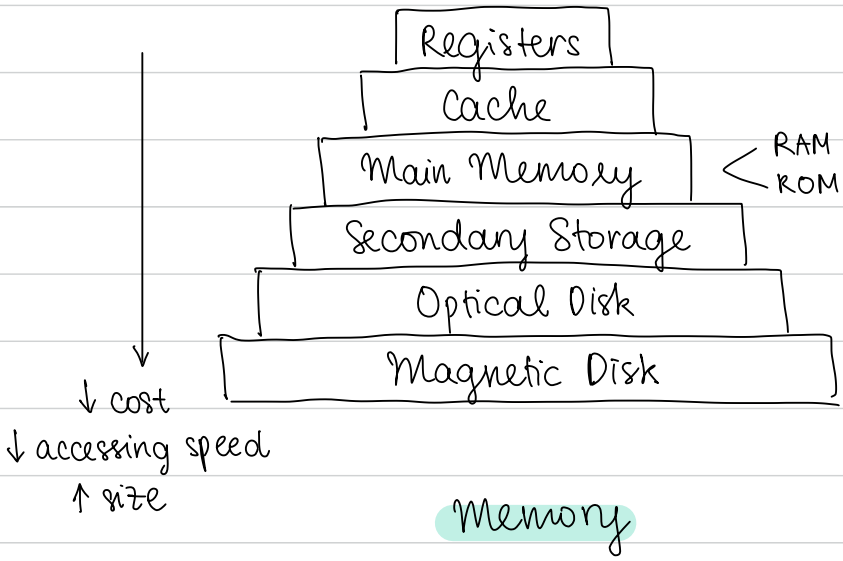
⑤ Notions of Operating system
- Process and Thread
- System's resources
- Shell
- System calls

⑥ Operating system's structure
- OS's components
- OS's services
- System calls
- System's structures

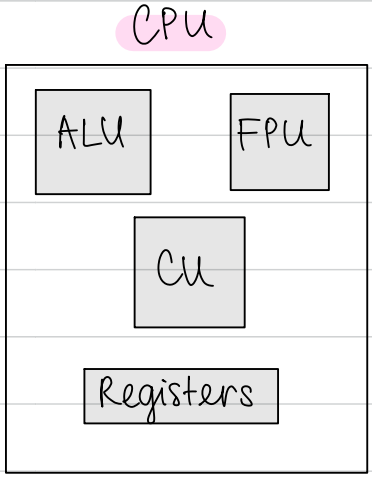
⑦ Principles of Operating System

# ① OS Concepts

## * Hardware




} fast : part of CPU




ALU : arithmetic-logic unit
FPU : floating-point unit
CU : control unit

### Registers

① General purposes

↳ can be used for lots of tasks

AX : accumulator
BX : base
CX : count
DX : data

(16-bit)

short

8-bit high

| AH | AL |
|---|---|

8-bit low

↓

similarly...

BH/BL CH/CL DH/DL

int

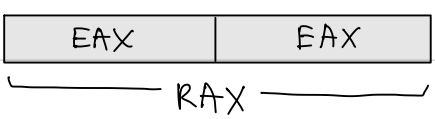
long

Extended (32-bit)

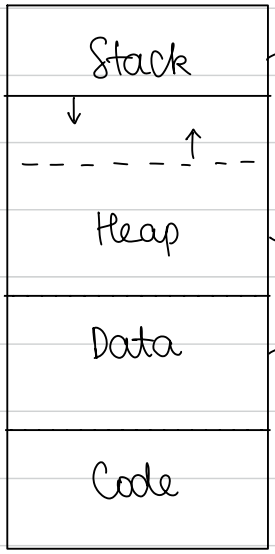
EAX
EBX
ECX
EDX

(64-bit)

RAX
RBX
RCX
RDX

## ② Program - specific ?




store parameters of a func
↑ size w/ ∀ func call

for dynamic alloc.

for static alloc. (store value init for variables)

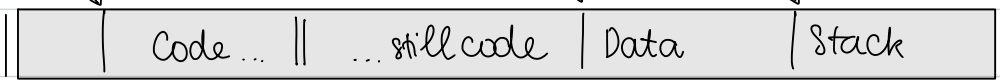
4 parts of a program.

SP : stack pointer (đỉnh của stack)

BP (phụ trợ)

store the address of the code seg.

Registers → CS (code seg) GS ES DS SS



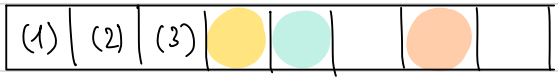

IP (instruction pointer)
Intel term ®

PC
(program counter)
(other manufacturer)

where/which instruction we've performed - to perform next

## ③ Flags (1 register for flag : 8 flags)

(overflow flag) OF ¬ (zero flag) ZF → use for comparison/ substraction → when result = 0



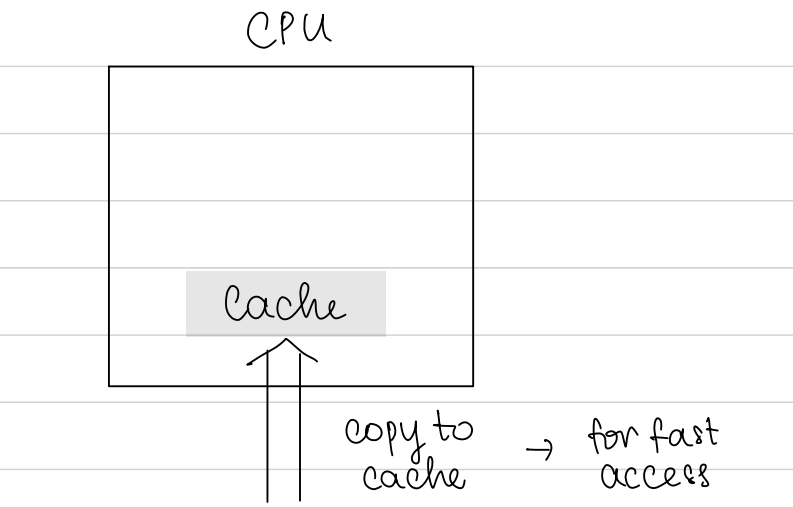

status

CF (carry flag)
→ for non-sign nb

TF (trap)
→ establish by compiler
→ for debugging

DF (direction)
→ direct the way of index traverse

## Cache




(recently used data)
↓
there is a high chance we'll reuse them

**Bus** (→ for transport ...)

(1) Address → address of data block CPU need
(2) Data → the data
(3) Control → control signal from proc. ↓ other parts

* **Software**

a) Commercial
① Licensed
② Free
③ Trial

b) Source-code Openness
① Open-src
② Closed

c) System-closeness

Application → provide services to the USERS directly

System Program → provide services so that other app can use the services to work
↳ serve APPS

OS
(∈ middle ware)

BIOS
(basic I/O system)
→ made by hardware manufacturer
→ basic programs that work w/ hardware
↳ store on ROM