# 

# 

# 

# Operating Systems Type:

Xử lý theo lô **(batch processing)**

* Các chương trình được đưa vào hàng chờ
* Máy tính thực hiện tuần tự các chương trình của người sử dụng
* Chương trình không có giao tiếp với người sử dụng

Đa chương trình **(multiprogramming)**

* Các chương trình được xếp hàng
* Một chương trình được thực hiện và chiếm giữ CPU cho đến khi (1) có yêu cầu vào/ra, hoặc (2) kết thúc
* Khi (1) hoặc (2) xảy ra, chương trình khác sẽ được thực hiện
* Tận dụng CPU tốt hơn xử lý theo lô đơn giản
* A subset of total jobs in system is kept in memory
* Gồm
  + **MultiTasking**
  + **Multi user** (Bao gồm Multi - tasking)

Phân chia thời gian **(time-sharing/multitasking)**

* the response time should be short

Hệ điều hành cho máy cá nhân **(Uniprograming)**

Xử lý song song **Parallel processing (PC-cluster)**

* search engines, e.g., google, yahoo,..

Thời gian thực **(real-time)**

Nhúng **(embedded) :**

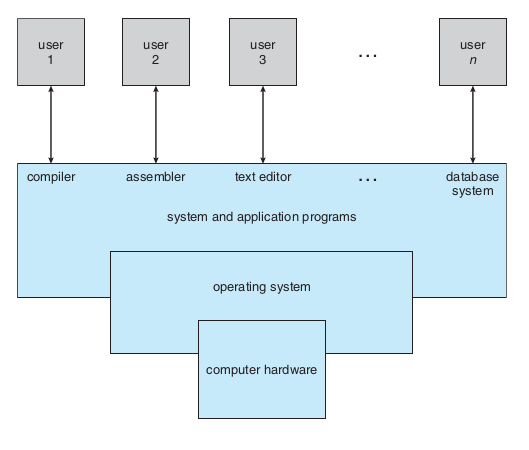
* embedded into devices to do a specific task)
* – the task has limited (few) functions
* – usually is made as a firmware (NOT as software)
* – Calculator, game players, digital camera, mp3 player, ...

Cầm tay **(portable)**

Đa phương tiện **(multimedia)**

Chuyên dụng **(special-purpose)**

# Thành phần MT:



The hardware—the central processing unit ( CPU ), the memory, and the

input/output ( I/O ) devices—provides the basic computing resources for the

system. The application programs—such as word processors, spreadsheets,

compilers, and Web browsers—define the ways in which these resources are

used to solve users’ computing problems.

# Mục đích **Operating Systems** :

Execute user programs and make solving user problems easier.

Make the computer system convenient to use.

Use the computer **hardware (resources**) in an efficient manner.

# Main tasks of an OS (Nhiệm vụ chính)

Process Management

• Memory Management (RAM)

• Storage Management

– File/directory

– Disks

• Protection and Security

• Networking

• Main tasks are usually implemented in kernel (core)

# Cấu trúc OS (OS structure)

**Layered approach**

– OS is divided into several levels

– A higher level can only access/use its direct lower level

• level 3 can access level 2

• level 4 cannot access

• level 2

**Microkernel**

Giữ cho nhân có các đủ các chức năng thiết yếu nhất để giảm cỡ

Các chức năng khác được đưa ra ngoài nhân

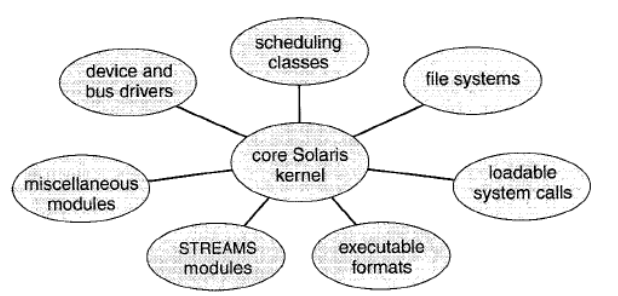
• Keep minimum/essential functions in kernel

• Others are built as libraries/applications

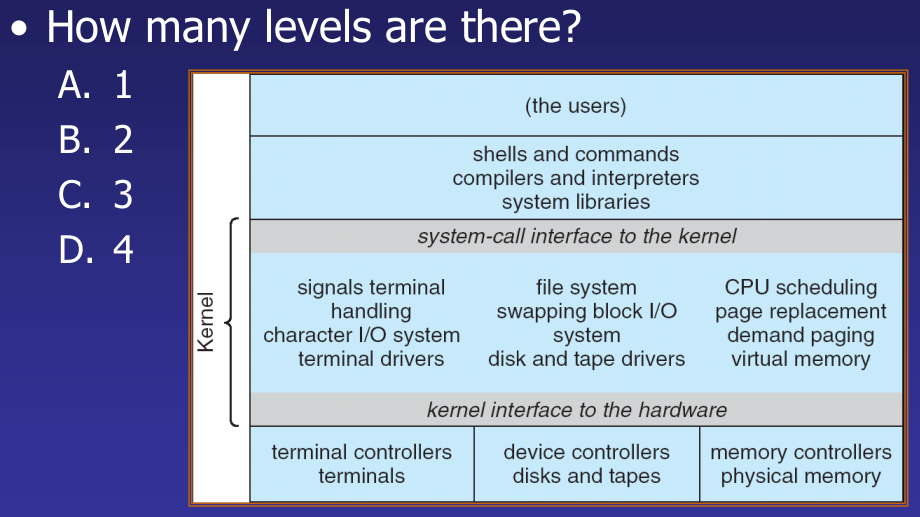
**Module approach**

Considered as most effective approach

– Inherits OOP paradigm



**UNIX**



**How many levels are there?**

A. 1

B. 2

C. 3

D. 4

# Phân biệt "Software > Application > Program > Process "

|  |  |
| --- | --- |
| 1. Software |  |
| 2. Application | On OS (Operate System) |
| 3. Program | * Compile * on Disk (second memory) or Storage |
| 4. Process | * Program & Runing on system * on RAM |

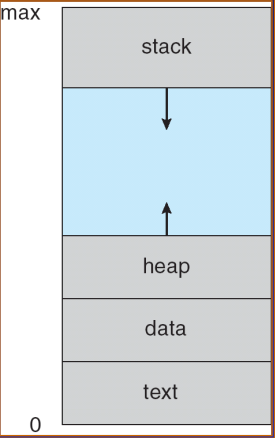
------------------------ Chương 2 ------------------------------

# PROCESS:

1. System processes

2. User processes

**Process structure**



**Process Control Block (PCB)**

– Process state

– Program counter

– CPU registers

– CPU scheduling information

– Memory-management information

– Accounting information

– I/O status information

NOTES: **Process structure & Process Control Block (PCB)** different among OSes( Khác nhau trên từng HĐH)

**Burst – a time span (duration)**

– IO burst

– CPU burst

**Context switch**

– CPU stops current process and runs another one

Progress steps

– save the state of the current process (Lưu trạng thái của A)

– put it into the **READY queue** (Đặt A lưu vào hàng đợi READY)

– pick the target process (Lấy B ra)

– restore the state of the target process (Khôi phục trạng thái B)

– run the target process (Chạy B)

NOTEs: **context swith** save in **READY queue**; I/O or event wait => in Waiting queue

**Queue:**

* Các tiến trình chưa được phân phối sử dụng CPU sẽ được đưa vào hàng chờ (queue)
* Có thể có nhiều hàng chờ trong hệ thống:
  + Hàng chờ sử dụng CPU, hàng chờ sử dụng máy in, hàng chờ sử dụng ổ đĩa CD…
* Trong suốt thời gian tồn tại, tiến trình phải di chuyển giữa các hàng chờ

SCHEDULING

• Long-term scheduler (or job scheduler)

– selects which processes should be brought into the ready queue

* Thường dùng trong các hệ xử lý theo lô
* Đưa tiến trình từ spool vào bộ nhớ trong

• Short-term scheduler (or CPU scheduler)

* selects which process should be executed next
* Lựa chọn tiến trình tiếp theo được sử dụng CPU

• Medium-term scheduler

* selects which process to temporarily swap out (of the MEM)
* Hay còn gọi là swapping (tráo đổi)
* Di chuyển tiến trình đang trong trạng thái chờ giữa bộ nhớ trong và bộ nhớ ngoài

**Dispatcher module**

gives control of the CPU to the process selected by the short-term

scheduler; this involves:

– switching context

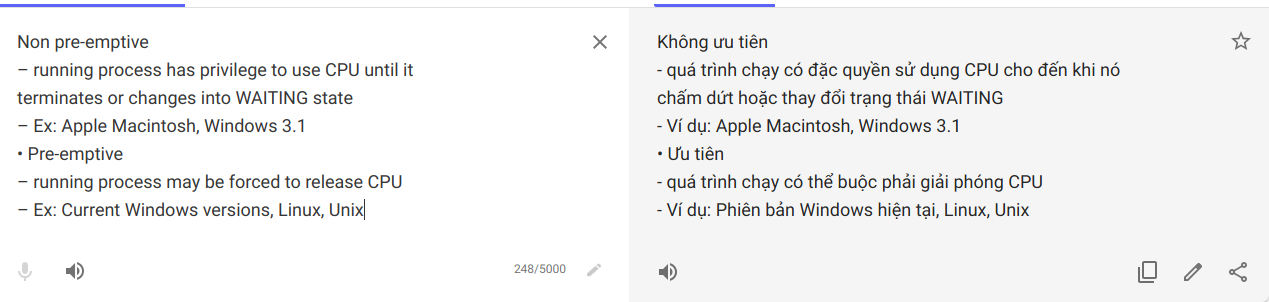
– switching to user mode

– run the process

• Dispatch latency – time it takes for the

dispatcher to stop one process and start

another running



**Multilevel Queue**

• Ready queue is partitioned into separate queues (nhiều hàng đợi)

– foreground (interactive)

– background (batch)

• Each queue has its own scheduling algorithm (mỗi hàng đợi 1 thuật toán)

– foreground – RR

– background – FCFS

Scheduling must be done between the queues

– Fixed priority scheduling (Cố định ưu tiên)

• (i.e., serve all from foreground then from

background)

• Possibility of starvation (khả năng chết đói)

– Time slice (lát tgian)

• each queue gets a certain amount of CPU time which

it can schedule amongst its processes

– i.e., 80% to foreground in RR

– 20% to background in FCFS

**Multiple-Processor Scheduling**

• CPU scheduling more complex when

multiple CPUs are available

– Homogeneous processors within a

multiprocessor

– Load sharing

• Asymmetric multiprocessing

– only one processor accesses the system data

structures, alleviating the need for data

sharing

