HỆ ĐIỀU HÀNH
(OPERATING SYSTEM)

Trình bày:Nguyễn Hoàng Việt
Khoa Công Nghệ Thông Tin
Đại Học Cần Thơ

Chương 3: Quá trình (Process)

- Khái niệm về quá trình
- Định thời cho quá trình
- Các thao tác trên quá trình
- Giao tiếp liên quá trình

Khái niệm về quá trình (1)
Định nghĩa

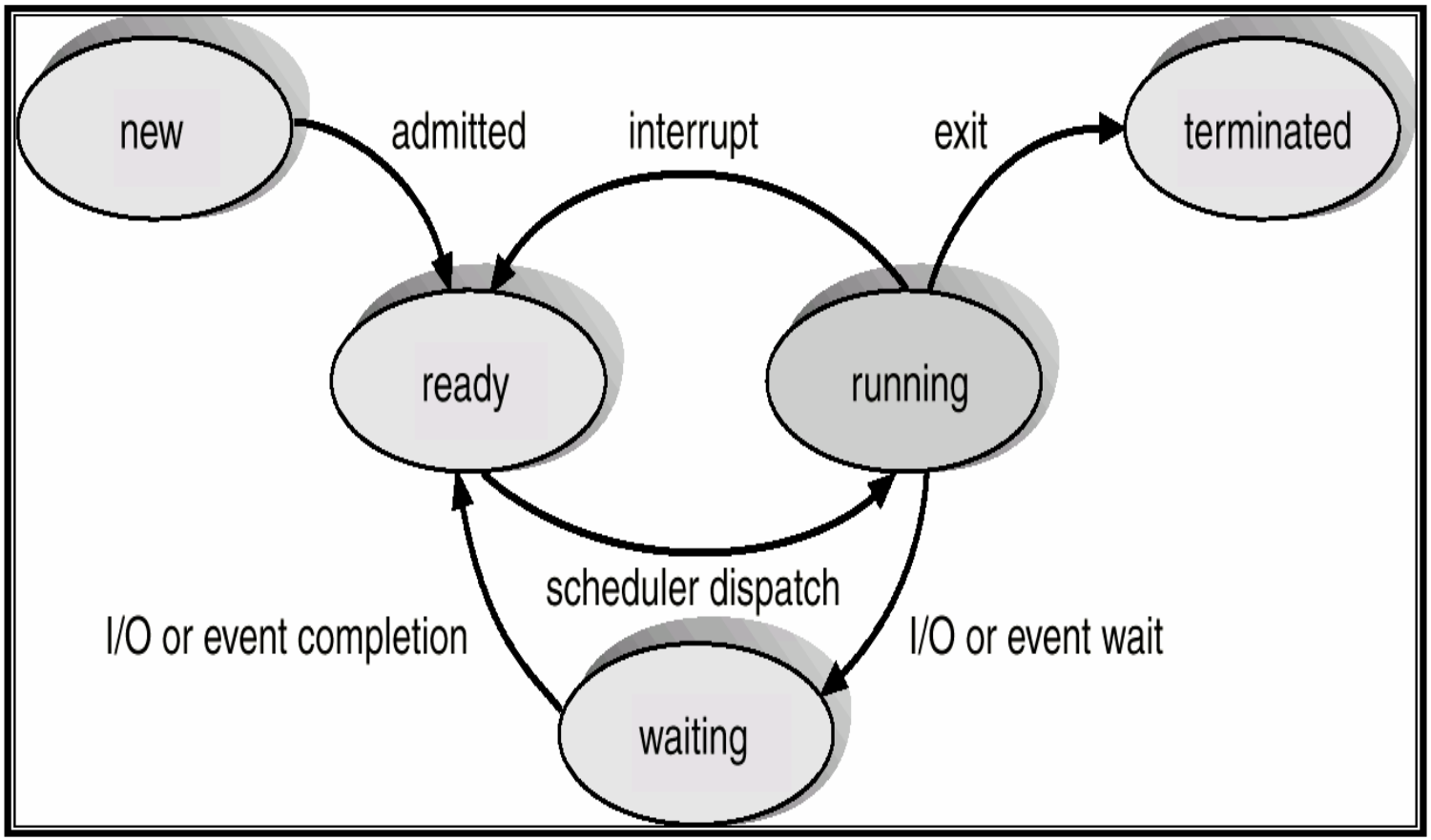
- Hệ điều hành có thể thực thi nhiều dạng chương trình:
  - Các chương trình HĐH – system calls
  - Hệ thống bó – các công việc (jobs)
  - Hệ thống chia thời gian - chương trình người dùng (user program) hay tác vụ (task)
  - Hệ thống một người dùng – các chương trình ứng dụng khác nhau
- Quá trình - là một chương trình đang thực thi. Sự thực thi của quá trình diễn ra theo cách thức tuần tự.
- Một quá trình bao gồm:
  - Mã lệnh chương trình (program code)
  - Bộ đếm chương trình (program counter) và các thanh ghi của CPU
  - Ngăn xếp (stack)
  - Phần dữ liệu (data section)
  - Có thể gồm phần bộ nhớ cấp phát động khi quá trình chạy (heap)

Khái niệm về quá trình (2)
Trạng thái của quá trình (Process state)

Một quá trình có thể có một trong các trạng thái sau:

- **new**: quá trình đang được khởi tạo.
- **running**: các chỉ thị của quá trình đang được thực thi.
- **waiting**: quá trình đang chờ đợi một sự kiện nào đó xuất hiện (hoàn thành xuất/nhập, chờ đợi một tín hiệu).
- **ready**: quá trình đang đợi để được sử dụng CPU.
- **terminated**: quá trình đã kết thúc.

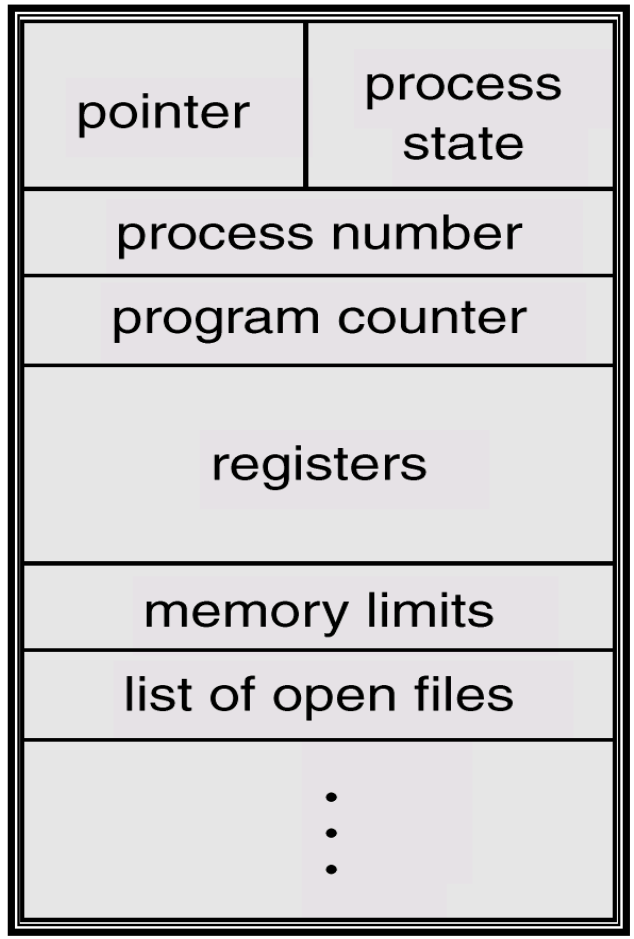
Khái niệm về quá trình (3)
Lưu đồ trạng thái của quá trình



Khái niệm về quá trình (4)
Khối điều khiển quá trình (PCB - Process Control Block)

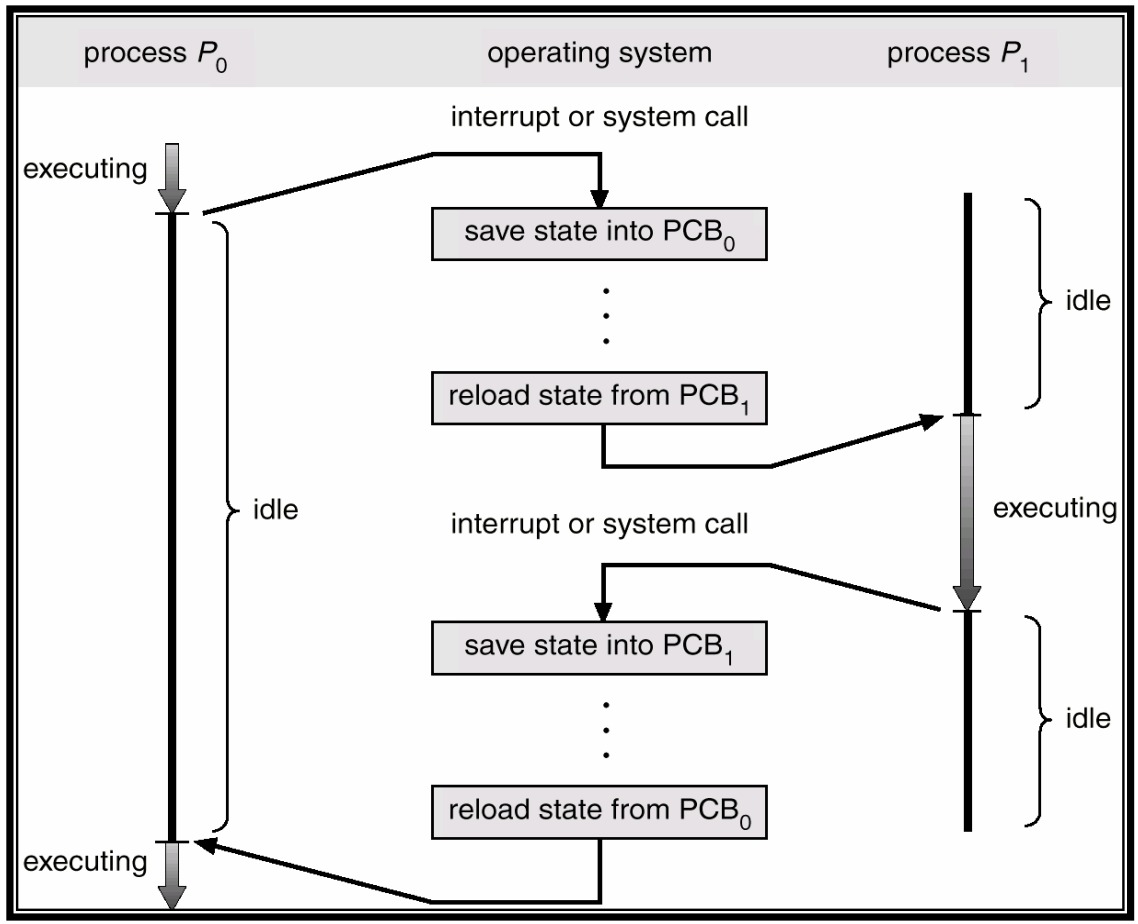
Là thông tin kết hợp với mỗi quá trình:

- Trạng thái của quá trình
- Bộ đếm chương trình
- Các thanh ghi
- Thông tin về định thời sử dụng CPU
- Thông tin về quản lý bộ nhớ
- Thông tin về chi phí
- Thông tin về trạng thái nhập/xuất



Khái niệm về quá trình (5)
Chuyển CPU phục vụ các quá trình

- PCB được xem như một nơi cất giữ các thông tin cho các quá trình.
- Thông tin trạng thái phải được lưu trữ khi một interrupt xuất hiện, nhằm cho phép quá trình có thể tiếp tục chính xác về sau.



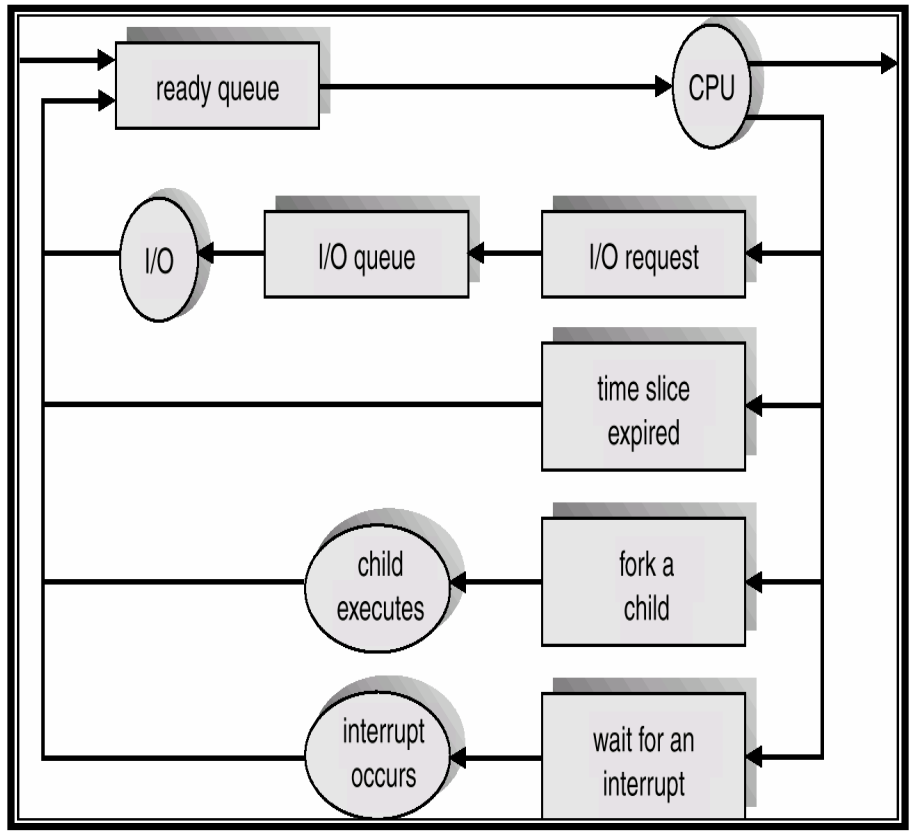
Khái niệm về quá trình (6)
Chuyển ngữ cảnh (Context Switch)

- Khi CPU chuyển sang một quá trình khác, hệ thống phải lưu lại trạng thái của quá trình cũ và nạp trạng thái đã lưu trước đây của quá trình. Tác vụ này gọi là context switch.
- Thời gian cho context switch là một phí tổn – hệ thống thực hiện công việc vô ích.
- Thời gian này phụ thuộc vào hỗ trợ của phần cứng:
  - Tốc độ chuyển phụ thuộc vào tốc độ bộ nhớ, số lượng thanh ghi phải được sao chép và các chỉ thị đặc biệt (như chỉ thị dùng nạp và lưu trữ tất cả các thanh ghi)
  - Tốc độ thông thường từ 1 đến 1000 µs
  - Thời gian này có thể là một thắt cổ chai (bottleneck) trong HĐH phức tạp

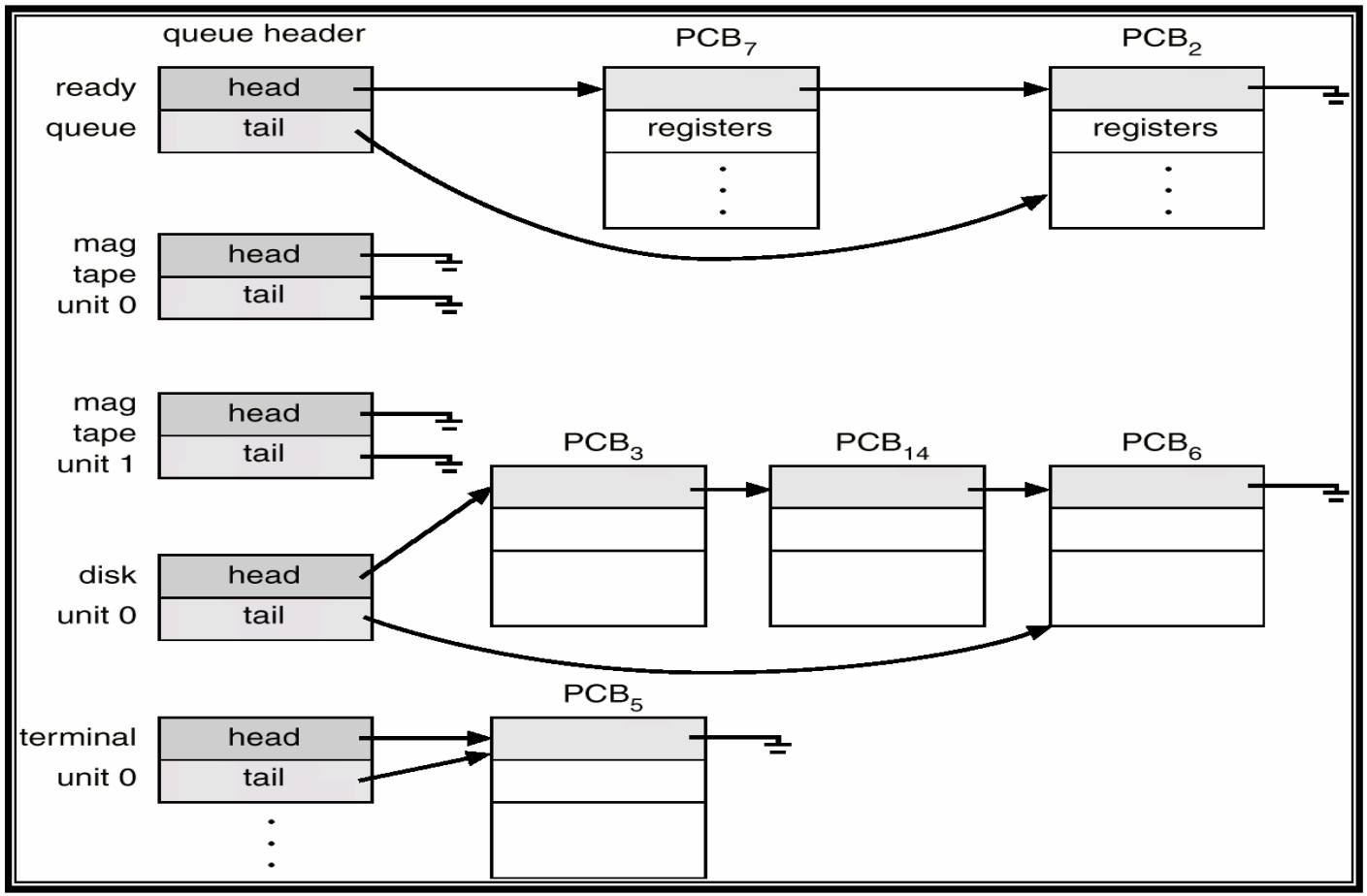
Định thời cho quá trình (1)
(Process Scheduling)

Các hàng đợi định thời:

- Hàng đợi công việc (Job queue): tập hợp tất cả các quá trình trong hệ thống.
- Hàng đợi sẵn sàng (Ready queue): tập hợp tất cả các quá trình đang nằm trong bộ nhớ, sẵn sàng và đang chờ để thực thi.
- Hàng đợi thiết bị (Device queue): tập hợp các quá trình đang đợi sử dụng một thiết bị xuất/nhập.
- Quá trình có thể di chuyển giữa các hàng đợi khác nhau.



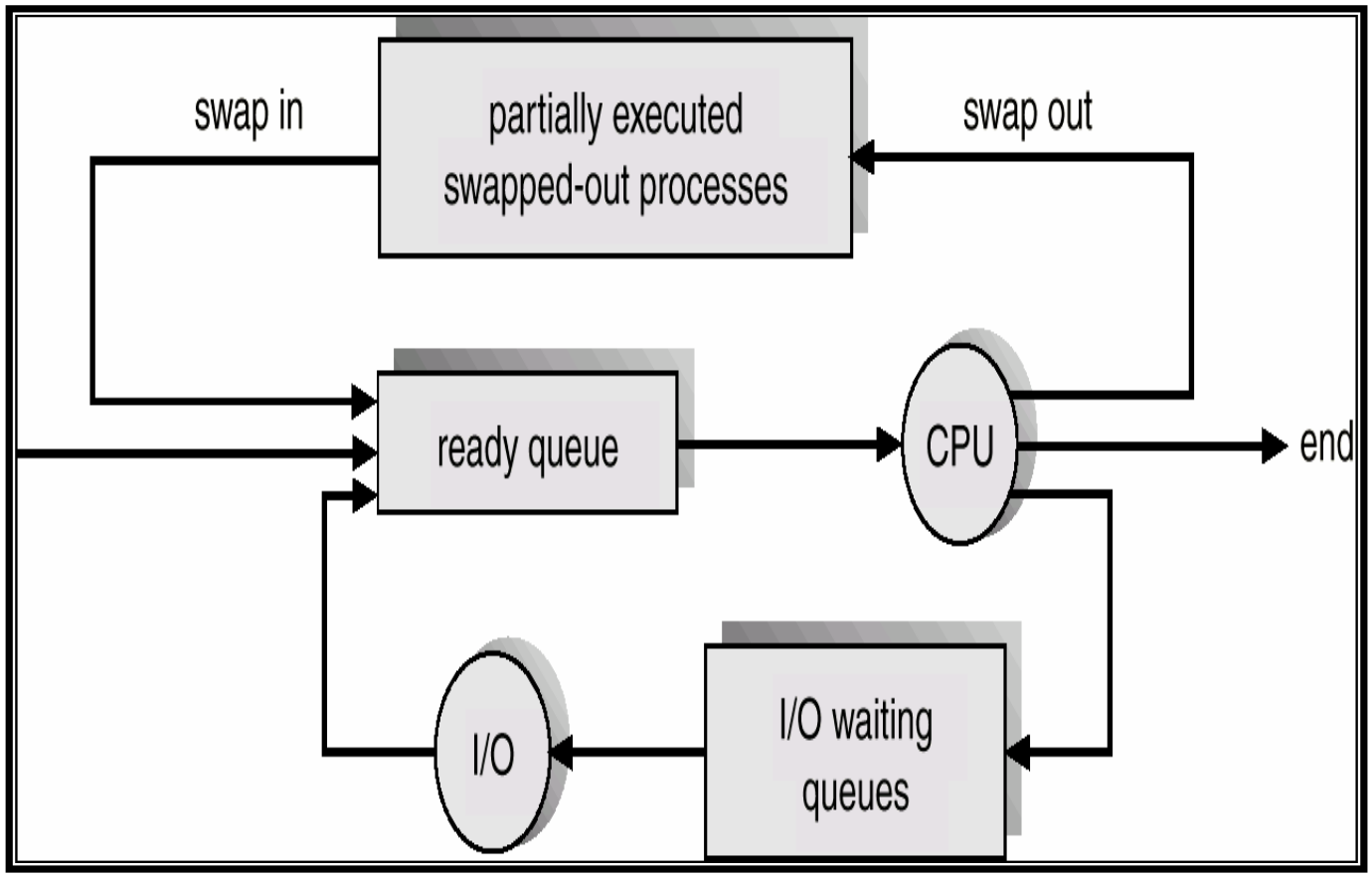
Định thời cho quá trình (2)
Hàng đợi sẵn sàng và nhiều hàng đợi thiết bị



Định thời cho quá trình (3)
Các bộ định thời

- Bộ định thời dài kỳ (Long-term scheduler/job scheduler) – chọn quá trình nào sẽ được đặt vào hàng đợi sẵn sàng (nạp vào bộ nhớ).
- Bộ định thời ngắn kỳ (Short-term scheduler/CPU scheduler) – chọn ra quá trình sẽ được thực thi kế tiếp và cấp CPU cho nó.
- Bộ định thời trung kỳ (Medium-term scheduling) – thực hiện hoán vị (swapping) các quá trình ra/vào bộ nhớ/đĩa do cạnh tranh CPU, bộ nhớ.

Định thời cho quá trình (4)
Bộ định thời trung kỳ



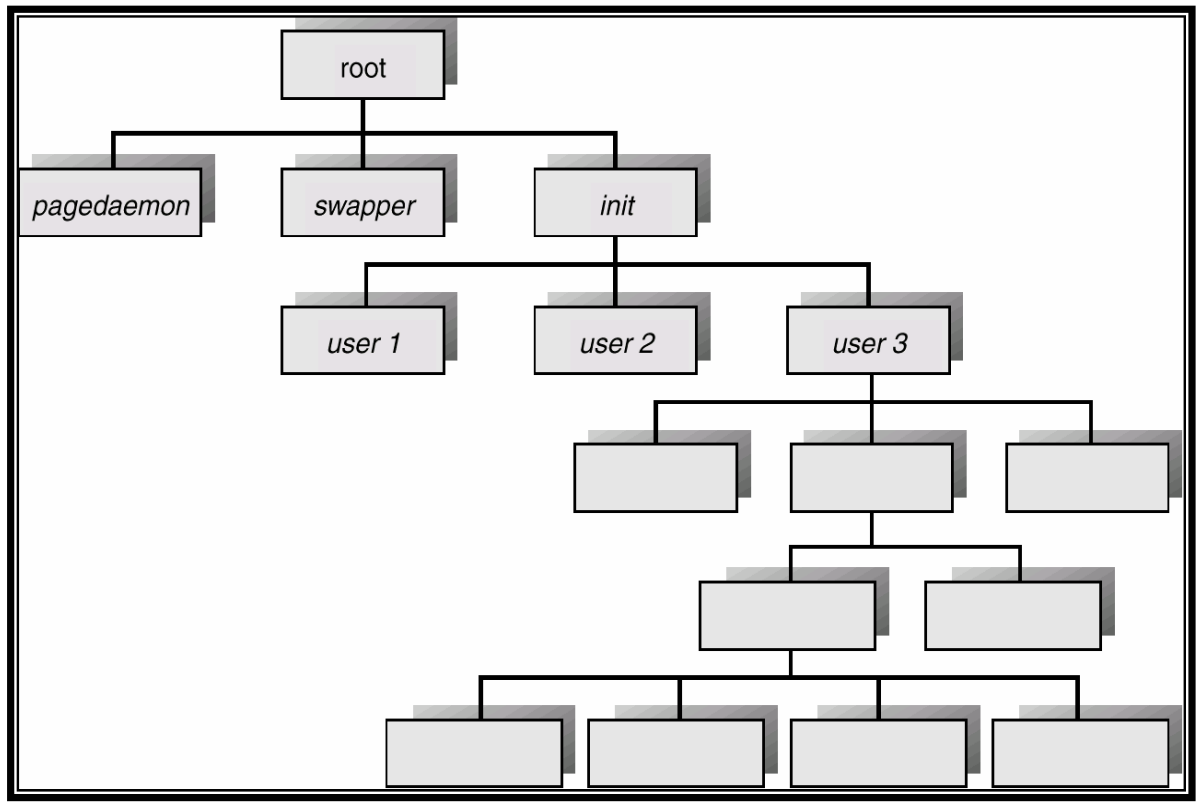
Định thời cho quá trình (5)
Các bộ định thời

- Bộ định thời ngắn kỳ được gọi rất thường xuyên (milliseconds) $\Rightarrow$ (phải nhanh).
- Bộ định thời dài kỳ được gọi rất không thường xuyên (seconds, minutes) $\Rightarrow$ (có thể chậm).
- Bộ định thời dài kỳ khống chế cấp độ đa chương (degree of multiprogramming).

Các thao tác trên quá trình (1)
Tạo quá trình

Quá trình cha tạo ra quá trình con, đến lượt quá trình con này lại tạo ra những quá trình khác, tạo nên cây quá trình.

Cây quá trình trong hệ thống UNIX.



Các thao tác trên quá trình (2)
Tạo quá trình (tt)

- Chia sẻ tài nguyên – có nhiều lựa chọn:
  - Quá trình cha và con chia sẻ tất cả tài nguyên.
  - Quá trình con chia sẻ một phần tài nguyên của quá trình cha.
  - Quá trình cha và con không chia sẻ tài nguyên nào cả.
- Dữ liệu khởi tạo:
  - Được chuyển theo từ quá trình cha sang con.
- Thực thi:
  - Quá trình cha và con thực thi song song.
  - Quá trình cha đợi đến khi quá trình con hoàn thành.
- Không gian địa chỉ:
  - Quá trình con sao chép không gian địa chỉ của quá trình cha (cùng chương trình và dữ liệu).
  - Quá trình con tự nạp chương trình riêng của nó.

Các thao tác trên quá trình (3)
Tạo quá trình (tt)

- UNIX:
  - **fork** – là lời gọi hệ thống dùng tạo quá trình mới
  - **execlp** – là lời gọi hệ thống được sử dụng sau **fork** bởi một trong 2 quá trình để thay thế không gian địa chỉ của quá trình đã gọi execlp bằng một quá trình mới.
- Windows NT: hỗ trợ cả hai mô hình
  - Quá trình con sao chép từ quá trình cha.
  - Quá trình cha xác định tên của một chương trình cho hệ điều hành nạp vào không gian địa chỉ của quá trình mới.

Các thao tác trên quá trình (4)
Minh họa lệnh fork trong UNIX

```
#include <stdio.h>
#include <unistd.h>
int main(int argc, char *argv[])
{
int pid;
  /* fork  another process */
  pid = fork();
  if (pid < 0) {   /* error occurred */
     fprintf(stderr, "Fork Failed");
     exit(-1);
   }
   else if (pid ==0) {   /* child process */
      execlp("/bin/ls", "ls", NULL);
   {
   else {  /* parent process will wait for the child to complete  */
      wait(NULL);
      printf("Child complete");
      exit (0);
    }
}
```

Các thao tác trên quá trình (5)
Kết thúc quá trình

- Quá trình thực hiện câu lệnh cuối cùng và yêu cầu HĐH xóa nó (dùng **exit)**.
  - Xuất dữ liệu từ quá trình con lên quá trình cha (dùng **wait**).
  - Tài nguyên của quá trình bị thu hồi lại bởi hệ điều hành.
- Quá trình cha có thể kết thúc sự thực thi của quá trình con (dùng **abort**).
  - Quá trình con đã vượt quá số tài nguyên được cấp.
  - Công việc giao cho quá trình con nay không còn cần thiết nữa.
  - Quá trình cha đang thoát.
    - ✓ Hệ điều hành không cho phép quá trình con tiến hành khi quá trình cha của nó kết thúc.
    - ✓ Sự kết thúc hàng loạt các quá trình con (cascading termination).

Các thao tác trên quá trình (6)
Hợp tác quá trình

- **Quá trình độc lập** không thể ảnh hưởng hoặc không bị ảnh hưởng bởi sự thực thi của quá trình khác.
- **Hợp tác quá trình** có thể ảnh hưởng hoặc bị ảnh hưởng bởi sự thực thi của quá trình khác.
- Thuận lợi của sự hợp tác quá trình:
  - Chia sẻ thông tin
  - Gia tăng tốc độ tính toán (nếu máy có nhiều CPU)
  - Module hóa
  - Tiện dụng
- Các bài toán hợp tác quá trình: người sản xuất – người tiêu thụ, bộ đọc – bộ ghi, …

Giao tiếp liên quá trình (1)
(IPC – Interprocess Communication)

- IPC cung cấp cơ chế cho phép giao tiếp và đồng bộ hoạt động của các quá trình không chia sẻ cùng không gian địa chỉ (có thể nằm trên các máy khác nhau).
  - Hữu ích cho môi trường phân tán (ví dụ chat trên www).
- IPC dùng hệ thống chuyển thông điệp (Message Passing System).
- Hệ thống chuyển thông điệp có thể được định nghĩa theo nhiều cách.

Giao tiếp liên quá trình (2)
Hệ thống chuyển thông điệp

- Hệ thống chuyển thông điệp cho phép các quá trình giao tiếp với nhau mà không cần tham khảo đến các biến dùng chung.
- Cung cấp ít nhất 2 thao tác (operation):
  - send (*message*) – kích thước thông điệp cố định/biến đổi.
  - receive(*message)*
- Nếu *P* và *Q* giao tiếp với nhau, chúng cần:
  - Thiết lập nối kết giao tiếp (communication link) giữa chúng;
  - Trao đổi thông điệp thông qua send/receive.
- Cài đặt nối kết giao tiếp:
  - Thuộc tính vật lý: bộ nhớ chia sẻ, bus phần cứng, …
  - Thuộc tính luận lý: giao tiếp trực hay gián tiếp, đối xứng hay bất đối xứng, bằng bản sao hay tham chiếu, kích thước thông điệp cố định hay biến đổi.

Giao tiếp liên quá trình (3)
Giao tiếp trực tiếp

- Các quá trình phải được đặt tên rõ ràng:
  - Send (*P*, *message*) – gởi thông điệp tới quá trình P.
  - Receive*(Q, message)* – nhận thông điệp từ quá trình Q.
- Thuộc tính của nối kết giao tiếp:
  - Các nối kết được thiết lập tự động;
  - Một nối kết kết hợp với chính xác một cặp quá trình;
  - Giữa mỗi cặp quá trình tồn tại chính xác một nối kết;
  - Nối kết có thể một hướng (unidirectional), nhưng thông thường là hai hướng (bi-directional);
  - Đối xứng trong việc đánh địa chỉ: 2 quá trình phải biết tên nhau để giao tiếp;
  - Thay đổi để thực hiện tính bất đối xứng trong đánh địa chỉ:
    - ✓ Send (*P*, *message*) – gởi thông điệp tới quá trình P.
    - ✓ Receive*(id, message)* – nhận thông điệp từ quá trình bất kỳ.

Giao tiếp liên quá trình (4)
Giao tiếp gián tiếp

- Các thông điệp được gởi và nhận thông qua hộp thư (mailbox), cũng được xem như cổng (port).
  - Mỗi mailbox có một định danh (id) duy nhất;
  - Các quá trình chỉ có thể giao tiếp nếu chúng dùng chung mailbox.
    - ✓ Send (*A*, *message*) – gởi thông điệp tới hộp thư A.
    - ✓ Receive*(A, message)* – nhận thông điệp từ hộp thư A.
- Thuộc tính của nối kết giao tiếp:
  - Nối kết chỉ được thiết lập nếu các quá trình chia sẻ một hộp thư chung;
  - Một nối kết có thể kết hợp với nhiều quá trình;
  - Mỗi cặp quá trình có thể dùng chung nhiều nối kết giao tiếp.
  - Nối kết có thể một hướng hay hai hướng.

Giao tiếp liên quá trình (5)
Giao tiếp gián tiếp

- Các thao tác:
  - Tạo hộp thư mới;
  - Gởi và nhận thông điệp thông qua hộp thư;
  - Xóa hộp thư.
- Chia sẻ hộp thư:
  - P_1, P_2, and P_3 chia sẻ hộp thư A.
  - P_1, gởi; P_2 and P_3 đang nhận từ A
  - Quá trình nào nhận thông điệp? Tùy vào chọn 1 trong các giải pháp.
- Giải pháp:
  - Cho phép một nối kết kết hợp với nhiều nhất 2 quá trình.
  - Cho phép chỉ 1 quá trình thực hiện thao tác nhận tại một thời điểm.
  - Cho phép hệ thống chọn một cách bất kỳ quá trình nhận (không phải cả hai). Bên gởi sẽ được thông báo ai đã nhận.

Giao tiếp liên quá trình (6)
Đồng bộ hóa (Synchronization)

- Chuyển thông điệp có thể nghẽn hoặc không nghẽn (blocking/non-blocking).
- Nghẽn được xem là đồng bộ, không nghẽn là không đồng bộ.
- Send và Receive có thể là nghẽn hoặc không nghẽn:
  - Send nghẽn: quá trình gởi bị nghẽn đến khi thông điệp được nhận.
  - Send không nghẽn: quá trình gởi gởi thông điệp và tiếp tục hoạt động.
  - Receive nghẽn: quá trình nhận nghẽn cho đến khi thông điệp sẵn dùng.
  - Receive không nghẽn: quá trình nhận nhận lại một thông điệp hợp lệ hay rỗng.

Giao tiếp liên quá trình (7)
Tạo vùng đệm (Buffering)

- Tạo hàng đợi để chứa các thông điệp liên quan đến một nối kết.
- Ba cách cài đặt:
  - Sức chứa là 0 (Zero capacity): quá trình gởi bị nghẽn đến khi thông điệp được nhận.
  - Sức chứa giới hạn (Bounded capacity): chiều dài giới hạn n thông điệp. Quá trình gởi nghẽn khi hàng đợi bị đầy.
  - Sức chứa không giới hạn (Unbounded capacity): chiều dài không giới hạn. Quá trình gởi không bao giờ nghẽn.

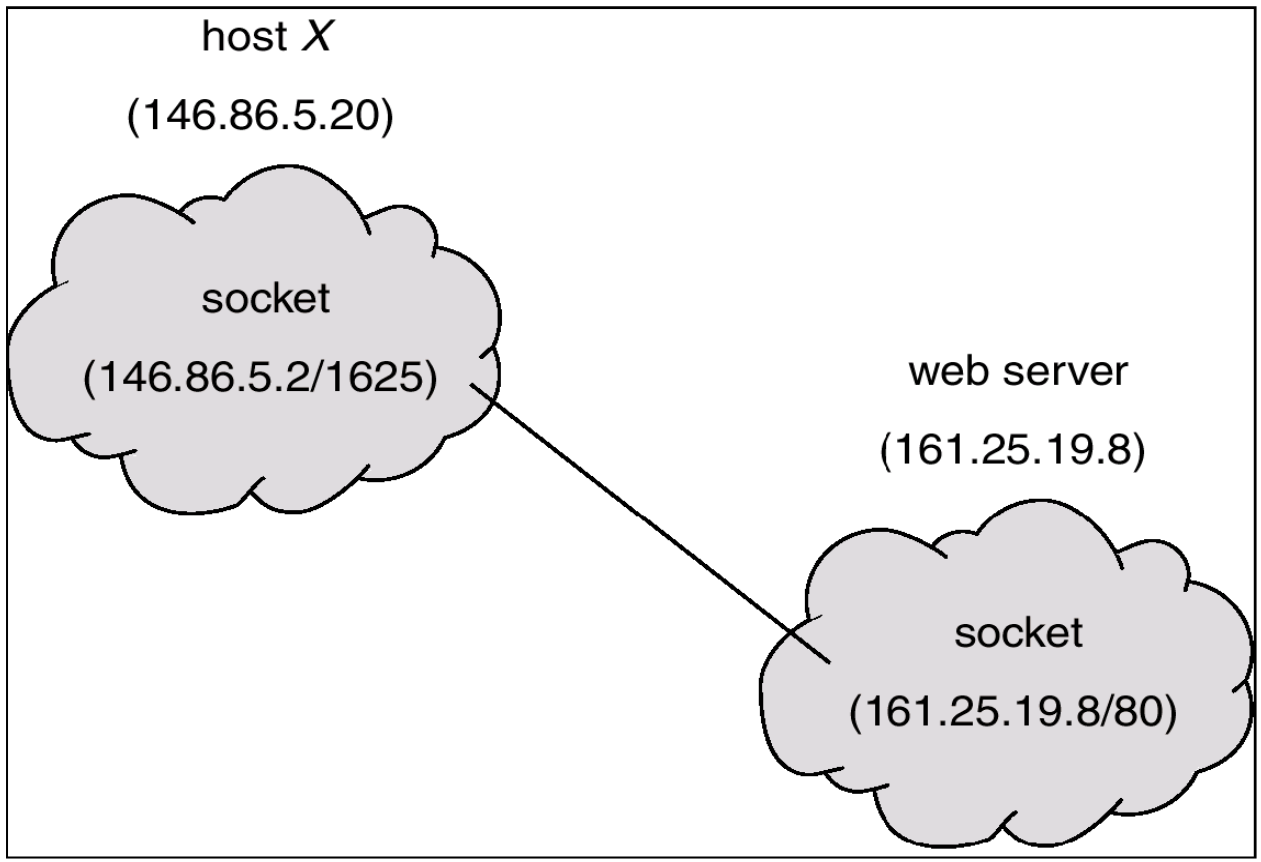
Giao tiếp khách-chủ (1)
(Client-server communication)

- Được dùng để truy cập dữ liệu trên các server.
- Các phương pháp shared memory và message passing cũng có thể được dùng cho giao tiếp khách-chủ
- Ba phương pháp khác thường dùng cho giao tiếp dạng này:
  - Sockets.
  - Remote Procedure Calls (RPCs)
  - Java's Remote Method Invocation (RMI)

Giao tiếp khách-chủ (1)
Sockets

- Một socket được định nghĩa như một điểm đầu cuối cho giao tiếp (endpoint for communication)
  - Một cặp quá trình giao tiếp qua mạng dùng một cặp socket – một cho mỗi quá trình
- Mỗi socket được hình thành từ 1 địa chỉ IP kết hợp với một số hiệu cổng (port number)
  - Socket **162.15.3.6:1625** là cổng **1625** trên máy **162.15.3.6**
- Các số hiệu cổng < 1024 được dành cho các dịch vụ chuẩn như telnet (23), ftp (21) và http (80)
- Khi một quá trình client khởi tạo một yêu cầu nối kết, nó được gán một cổng bởi máy
  - Tất cả nối kết phải duy nhất với mỗi quá trình có một số hiệu cổng khác nhau

Giao tiếp khách-chủ (2)
Sockets (tt)



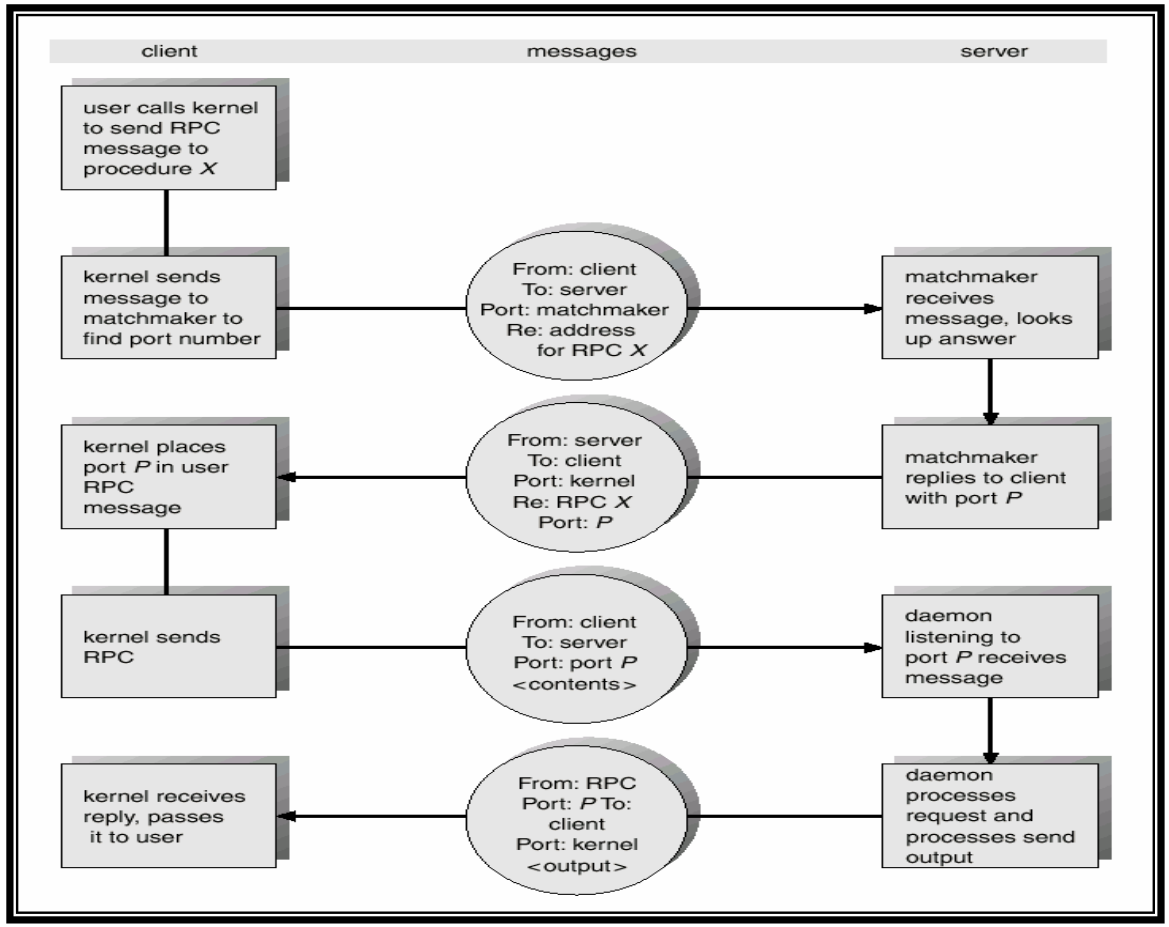
Socket communication

Giao tiếp khách-chủ (3)
Remote procedure Call (RPC)

- RPC trừu tượng hóa các lời gọi thủ tục (procedure calls) giữa các quá trình trên hệ thống mạng
  - Nó cho phép một client gọi một thủ tục trên một máy ở xa như nó gọi một thủ tục cục bộ
  - Message trao đổi trong giao tiếp RPC được sắp xếp có cấu trúc và được chuyển đến một RPC daemon lắng nghe trên một cổng tại máy ở xa.
- Stub: đại diện phía client cho một thủ tục thực trên server
  - Khi client gọi một thủ tuc ở xa, hệ thống RPC sẽ gọi stub tương ứng, chuyển cho nó các tham số được cung cấp tới thủ tuc ở xa
  - Stub này sẽ định vị cổng trên server và sắp xếp các tham số theo một định dang (đóng gói) và gởi nó đi bằng message passing
- Một stub tương tự trên server nhận message này, mở gói các tham số và gọi thực hiện thủ tục trên server

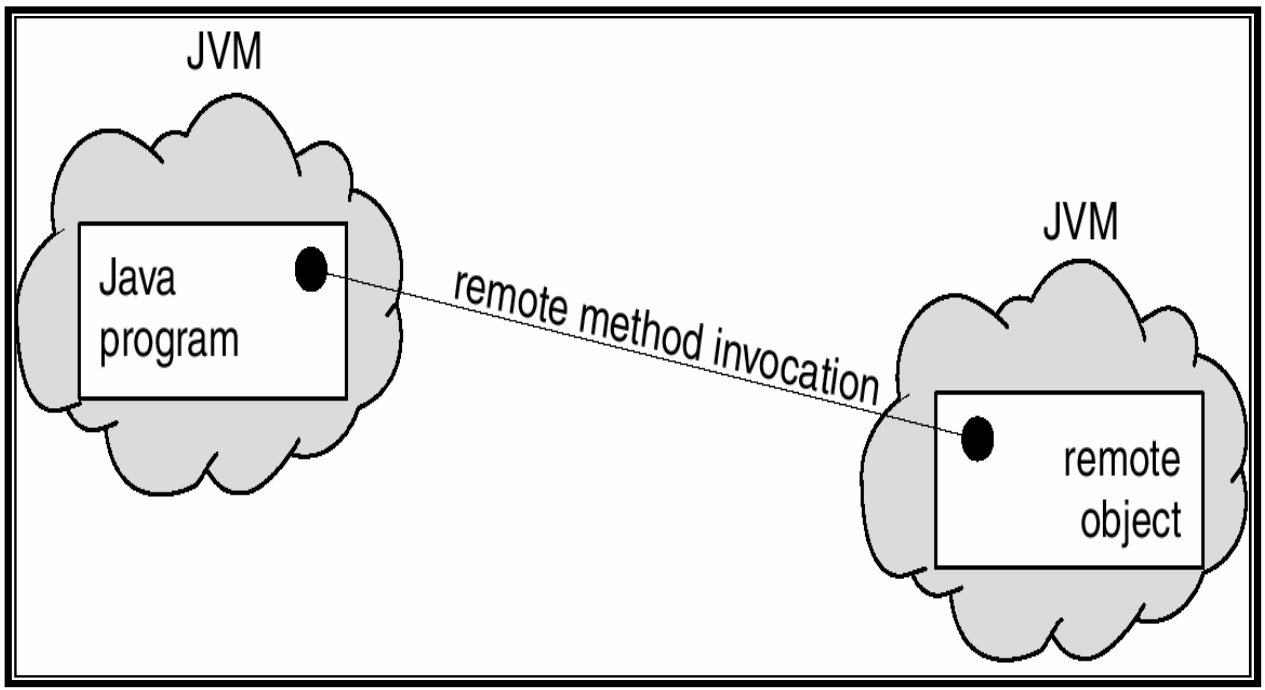
Giao tiếp khách-chủ (4)
Remote procedure Call (tt)

Thực thi của RPC



Giao tiếp khách-chủ (5)
Remote Method Invocation (RMI)

- RMI là một cơ chế Java tương tự RPC
- RMI cho phép một chương trình Java trên một máy gọi thực hiện một method trên một remote object



Giao tiếp khách-chủ (6)
Remote Method Invocation (tt)

- RMI cài đặt remote object dựa vào **stub** và **skeleton**.
- Stub là đại diện cho remote object trên phía client
- Skeleton đại diện cho remote object phía server