CHƯƠNG III. TRUYỀN THÔNG (5 LT + 2 BT)

I. Các thuật toán
II. Các kỹ thuật truyền thông
III. Các dịch vụ hỗ trợ
IV. Một số kiến trúc tiêu biểu

I. CÁC THUẬT TOÁN

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán

1. Mở đầu

- Thuật toán phân tán (distributed algorithm): thiết kế để chạy trên các thiết bị phần cứng của các máy tính kết nối với nhau.
- Thách thức: Tích phối thành công các hành vi của các phần độc lập của giải thuật khi
  - Các bộ xử lý xảy ra lỗi
  - Các kết nối truyền thông không tin cậy

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán

1. Mở đầu

- Nguyên tắc chung: đưa vào một số ràng buộc liên quan đến các sự kiện diễn ra theo giải thuật
- Có nhiều kiểu ràng buộc
  - Thứ tự các sự kiện (các kiểu đồng hồ đồng bộ khác nhau)
  - Ràng buộc về topology truyền thông (trao đổi thông điệp) của ứng dụng:

1. Mở đầu

- Các bài toán tiêu biểu:
  - **Bầu chọn người đứng đầu (leader election)**
  - Liên ứng (consensus)
  - Tìm kiếm phân tán (distributed search)
  - Sinh cây bắc cầu (spanning tree generation)
  - **Loại trừ lẫn nhau (mutual exclusion)**
  - Cấp phát tài nguyên (resource allocation)
  - Kết thúc (terminaison)
- Các lĩnh vực ứng dụng:
  - Viễn thông
  - Tính toán khoa học
  - Xử lý thông tin phân tán
  - Điều khiển tiến trình thời gian thực

GIANG

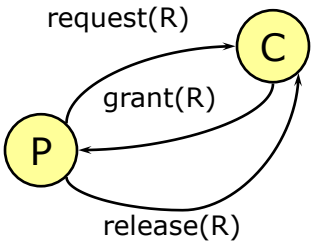
Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2.Thuật toán loại trừ lẫn nhau (mutual exclusion)
3.Thuật toán bầu cử (election)
4.Thuật toán kết thúc (terminaison)

a. Đặt vấn đề

- Đồng bộ các tiến trình (Process Synchronization): kỹ thuật để phối hợp việc thực hiện các tiến trình
  - 1 tiến trình có thể phải đợi các tiến trình khác
  - Việc dùng chung tài nguyên có thể đòi hỏi loại trừ (exclusion) 1 tiến trình truy nhập vào 1 tài nguyên tại 1 thời điểm
- Distributed mutual exclusion: tránh sử dụng đồng thời vào 1 tài nguyên
  - Bằng phần cứng: ngắt, cơ chế test & set
  - Bằng phần mềm: 1 đoạn mã CT (critical section)
    - Cho phép 1 tiến trình được truy cập 1 mình vào 1 tài nguyên
    - Giả thiết là các tiến trình đã thống nhất về cách định danh tài nguyên (truyền ID bằng thông điệp truy vấn)

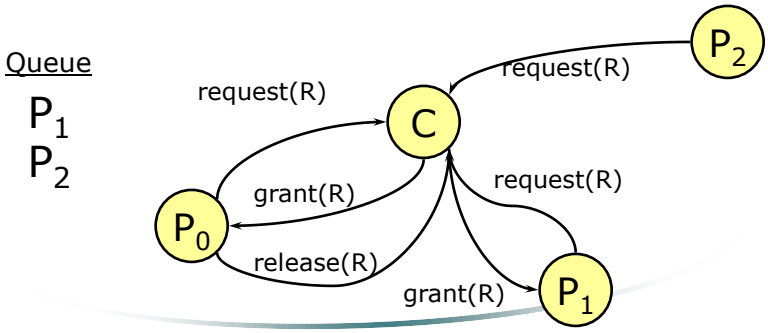
b. Giải thuật tập trung

- Phỏng theo hệ chỉ có 1 bộ xử lý
- 1 tiến trình được bầu làm tích phối viên (coordinator)



b. Giải thuật tập trung

- Nếu 1 tiến trình khác yêu cầu dùng tài nguyên:
  - Tích phối viên không trả lời cho đến khi giải phóng tài nguyên
  - Duy trì hàng đợi
    - Các yêu cầu dịch vụ theo thứ tự FIFO



GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
a. Đặt vấn đề
b. Giải thuật tập trung

b. Giải thuật tập trung

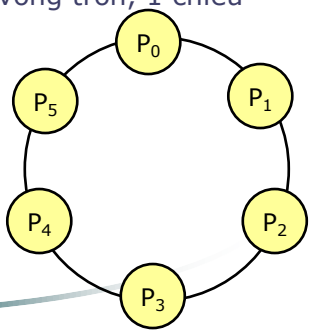
- Ưu điểm
  - Công bằng: mọi yêu cầu đều được xử lý theo thứ tự
  - Dễ cài đặt, dễ hiểu, dễ kiểm tra
- Vấn đề còn tồn tại
  - Các tiến trình không thể phân biệt được yêu cầu của mình đang bị đóng băng hay tiến trình tích phối viên chết
  - Nguy cơ tắc nghẽn (bottleneck)

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
a. Đặt vấn đề
b. Giải thuật tập trung
c. Giải thuật Token Ring

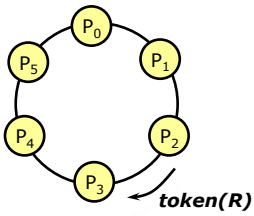
c. Giải thuật Token Ring

Giả sử biết trước 1 nhóm tiến trình
- Đặt thứ tự sử dụng cho nhóm đó (tùy chọn)
- Viết đoạn mã CT thể hiện vòng ảo (logical ring)
- Các tiến trình chỉ giao tiếp với tiến trình lân cận (neighbor) theo vòng tròn, 1 chiều



c. Giải thuật Token Ring

- Khởi tạo
  - Tiến trình 0 được phát thẻ (token) để sử dụng tài nguyên R
- Thẻ này được quay vòng
  - Từ tiến trình P_i đến tiến trình $P_{(i+1)}$mod N
- Khi tiến trình nhận được thẻ
  - Kiểm tra xem nó có cần sử dụng tài nguyên hay không (có tham gia vào critical section hay không)
  - Nếu không, chuyển token cho tiến trình hàng xóm
  - Nếu có, truy nhập vào tài nguyên
    - Giữ thẻ cho đến khi dùng xong



c. Giải thuật Token Ring

- Tại 1 thời điểm, chỉ có duy nhất 1 tiến trình có thẻ
  - Đảm bảo nguyên tắc loại trừ lẫn nhau
- Giữ đúng thứ tự
  - Không xảy ra tình trạng 1 tiến trình nào đó không bao giờ nhận được thẻ
- Nếu mất thẻ (tiến trình chết)
  - Phải sinh lại thẻ
- Không đảm bảo được thứ tự FIFO
  - Đôi khi đây là nhược điểm

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
a. Đặt vấn đề
b. Giải thuật tập trung
c. Giải thuật Token Ring
d. Giải thuật Ricard & Agrawala

d. Giải thuật Ricart & Agrawala

- Sử dụng đồng hồ logic và kỹ thuật truyền thông đa điểm tin cậy (reliable multicast)
- Tiến trình muốn tham dự vào critical section:
  - Soạn thông điệp chứa:
    - Identifier (machine ID, process ID)
    - Name of resource
    - Timestamp (totally-ordered Lamport)
  - Gửi yêu cầu tới tất cả các tiến trình trong nhóm
  - Chờ cho tới khi tất cả cho phép
  - Tham gia vào critical section / sử dụng tài nguyên

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
a. Đặt vấn đề
b. Giải thuật tập trung
c. Giải thuật Token Ring
d. Giải thuật Ricard & Agrawala

d. Giải thuật Ricart & Agrawala

- Khi tiến trình nhận được yêu cầu:
  - Nếu bên nhận không quan tâm: gửi tín hiệu OK cho bên gửi
  - Nếu bên nhận tham gia vào critical section: không trả lời, thêm yêu cầu vào hàng đợi
  - Nếu bên nhận cũng vừa mới gửi yêu cầu:
    - So sánh timestamps của các thông điệp gửi và nhận: ai gửi sớm hơn sẽ thắng
    - Nếu bên nhận thua, gửi tín hiệu OK
    - Nếu bên nhận thắng, không trả lời, thêm yêu cầu vào hàng đợi
- Khi xong critical section: gửi tín hiệu OK cho tất cả các bên có yêu cầu trong hàng đợi

→ N điểm gây lỗi, lượng thông điệp lớn

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)

Bài tập về nhà

- SV tự tìm hiểu các giải thuật sau đây
  - Giải thuật Lamfort
  - Giải thuật Suzuki-Masumi
  - Giải thuật Raymond
- Nêu tóm tắt các đặc điểm của từng giải thuật, so sánh số lượng các thông điệp cần truyền để thực hiện giải thuật

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
3. Thuật toán bầu cử (election)

a. Đặt vấn đề

- Cho 1 tập các tiến trình
  - Các tiến trình có đặc điểm tương tự nhau
  - Mỗi tiến trình chỉ có 1 ID duy nhất
- Cần xác định 1 và chỉ 1 tiến trình đóng vai trò tích phối viên

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
3. Thuật toán bầu cử (election)

b. Giải thuật Bully

- Chọn tiến trình có ID cao nhất làm tích phối viên
- Khi tiến trình P phát hiện tích phối viên chết:
  - Gửi thông điệp bầu cử đến tất cả các tiến trình có ID cao hơn
  - Tùy chọn: cho tất cả các nút có ID thấp hơn biết rằng đang tiến hành bầu cử
- Nếu tiến trình nhận được thông điệp bầu cử
  - Đáp lại bằng thông điệp OK
  - Tiến hành bầu cử (trừ trường hợp đã có tiến trình khác làm việc này)

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
3. Thuật toán bầu cử (election)

b. Giải thuật Bully

- Tiến trình A thông báo thắng cử bằng cách gửi đến tất cả các tiến trình khác 1 thông điệp nói rằng A là tích phối viên mới
- Nếu 1 tiến trình chết đi rồi sống lại, nó tiến hành bầu cử để tìm ra tích phối viên.

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
3. Thuật toán bầu cử (election)

c. Giải thuật vòng

- Các tiến trình được tổ chức thành vòng
- Nếu bất kỳ tiến trình nào phát hiện lỗi của tích phối viên
  - Gửi thông điệp bầu cử chứa ID của nó và gửi cho tiến trình hàng xóm tiếp theo
  - Nếu tiến trình tiếp theo chết thì bỏ qua
  - Lặp lại cho đến khi định vị được 1 tiến trình đang chạy
- Đến khi nhận được 1 thông điệp bầu cử
  - Tiến trình gửi chuyển tiếp thông điệp này, và thêm ID của nó vào phần thân thông điệp
- Đôi khi thông điệp bầu cử quay về chính tiến trình gửi nó đi
  - Tiến trình này sẽ phát hiện ra ID của nó trong danh sách ID nằm trong thông điệp
  - Gửi thông điệp thông báo cho các tiến trình khác nó chính là tích phối viên

GIANG VÝ

Chương III. Truyền thông
I. Các thuật toán
1. Mở đầu
2. Thuật toán loại trừ lẫn nhau (mutual exclusion)
3. Thuật toán bầu cử (election)

Bài tập về nhà

- Xác định các vấn đề có khả năng xảy ra trong khi bầu cử

II. CÁC KỸ THUẬT TRUYỀN THÔNG

GIANG

I. Các thuật toán
II. Các kỹ thuật truyền thông
1. **Mở đầu**
2. Gọi thủ tục từ xa
3. Truyền thông hướng thông điệp
4. Truyền thông hướng dòng
5. Truyền thông đa điểm

a. Truyền thông trong hệ phân tán

- Điều khiển bởi các giao thức phân tầng (layered protocols)
  - Đặc điểm của việc phân tầng?
  - Ví dụ? Dịch vụ hàng không và dịch vụ ăn uống.
    - Quyết định phục vụ thức ăn gì cho hành khách: manager.
    - Quyết định làm thế nào để đặt thực đơn này: secretary.
- Các tầng thấp hơn thực hiện các chức năng "cơ bắp" hơn.
- Cần có đặc tả chức năng của mỗi tầng và nhất trí giữa các tầng.
- 2 loại giao thức cơ bản: connection-oriented và connectionless.
  - Connection oriented cần làm 1 số thao tác để thiết lập 1 "virtual circuit" trước khi giao tiếp. Connectionless không cần làm việc này.
  - Examples?
  - Thuận lợi và khó khăn?

b. OSI (Open Systems Interconnection)7-Layer Stack

- Mỗi tầng chuẩn bị thông điệp và gửi cho tầng dưới.
  - Quy trình được đảo ngược khi nhận thông điệp

Application layer

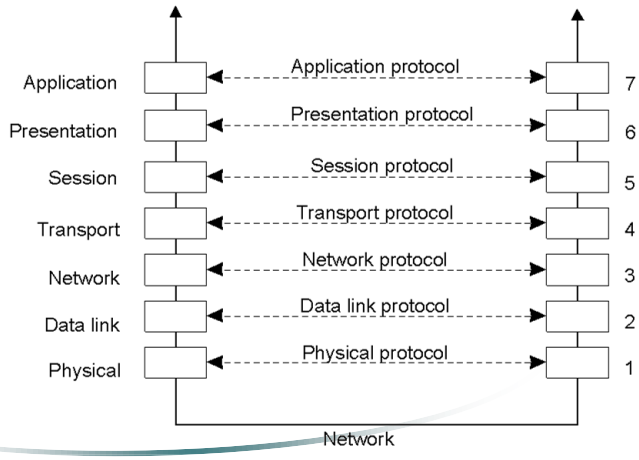
Protocols for things like mail, file transfer, communication terminals.

Presentation layer

Meaning of bits, define records, etc.

Session layer

Dialogs, checkpoints for long transfers



b. OSI 7-Layer Stack

- Mỗi tầng chuẩn bị thông điệp và gửi cho tầng dưới.
  - Quy trình được đảo ngược khi nhận thông điệp

Transport layer

Reliability

Network layer

Routing

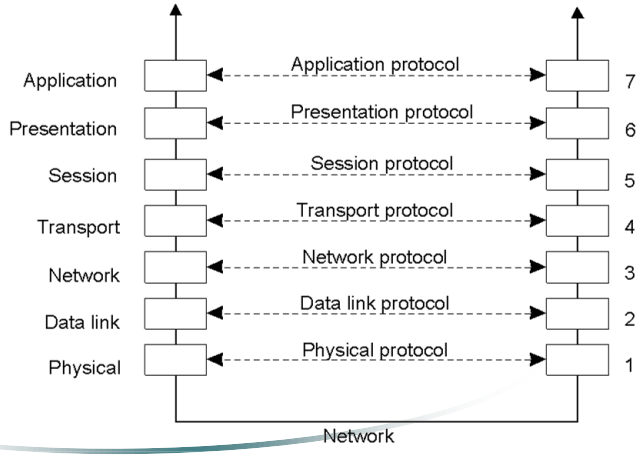
Data Link Layer

Physical layers are unreliable

Main job of this layer is to detect and correct errors (by check sum)

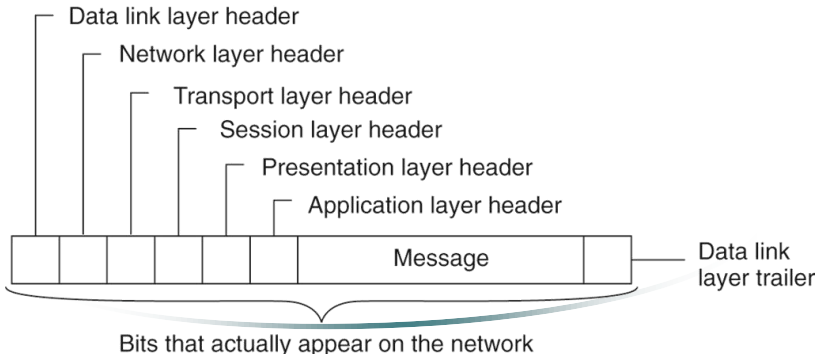
Physical Layer

Concerned with transmitting the 0s and 1s



c. Message Headers/Trailers

- Mỗi tầng sẽ thêm vào thông điệp phần đầu (header) hoặc đuôi (trailer) của riêng nó.
  - Có gì trong header/trailer?
- Khi nào thì header tốt hơn trailer?
- Khi nào trailer tốt hơn header?

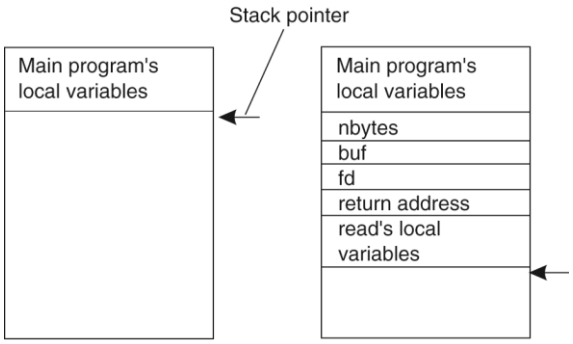


GIANG

I. Các thuật toán
II. Các kỹ thuật truyền thông
1. Mở đầu
2. **Gọi thủ tục từ xa**

a. Gọi thủ tục theo quy ước

- Xét count = read(fd, buf, nbytes).



Truyền tham số trong lời gọi thủ tục tại chỗ: bộ nhớ stack trước khi thực hiện lời gọi

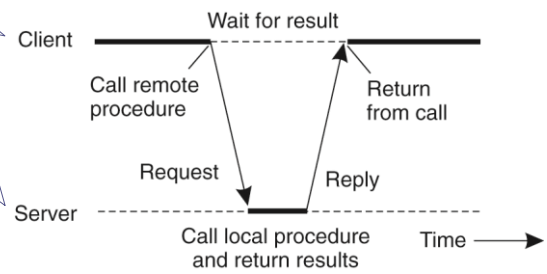
Bộ nhớ stack khi đang thực hiện thủ tục được gọi

b. Remote Procedure Call

- Thực hiện lời gọi thủ tục thông thường trên máy ở xa
  a = my_func(i, x);
- Cần làm những gì ?

Đóng gói các tham số + các thông tin điều khiển vào thông điệp (marshalling). Gửi thông điệp để xử lý trên máy ở xa

Mở thông điệp trên máy ở xa (unmarshalling). Gọi thủ tục tương ứng trên máy đó (gọi tại chỗ) Lấy kết quả trả về và gửi lại cho bên gọi.

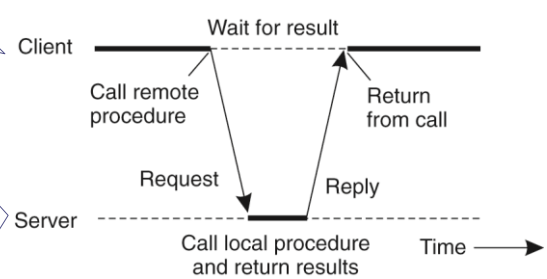


b. Remote Procedure Call

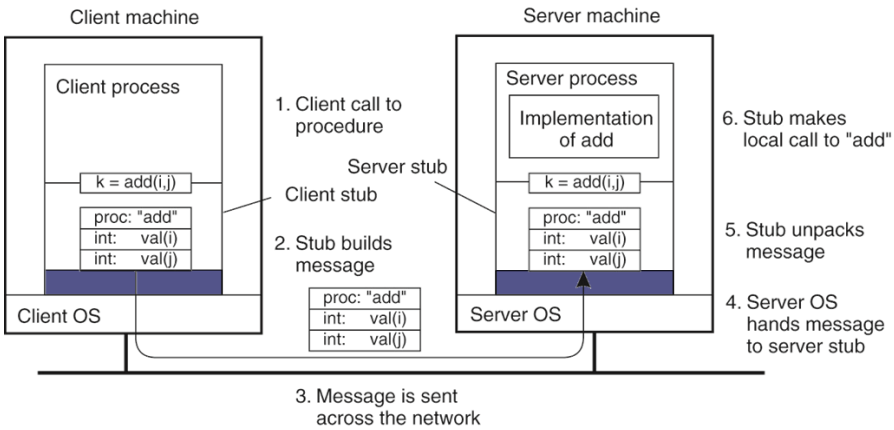
- Thực hiện lời gọi thủ tục thông thường trên máy ở xa
  a = my_func(i, x);
- Cần làm những gì ?

Client stub
Đoạn mã CT chịu trách nhiệm biến lời gọi từ xa thành lời gọi tại chỗ, và đóng gói lời gọi thành thông điệp

Server stub (skeleton)
Đoạn mã CT chịu trách nhiệm mở gói trên server, và gọi thủ tục trên chính server đó.
Lưu ý: việc thực hiện thủ tục không phải là trách nhiệm của stub



b. Remote Procedure Call



GIANG

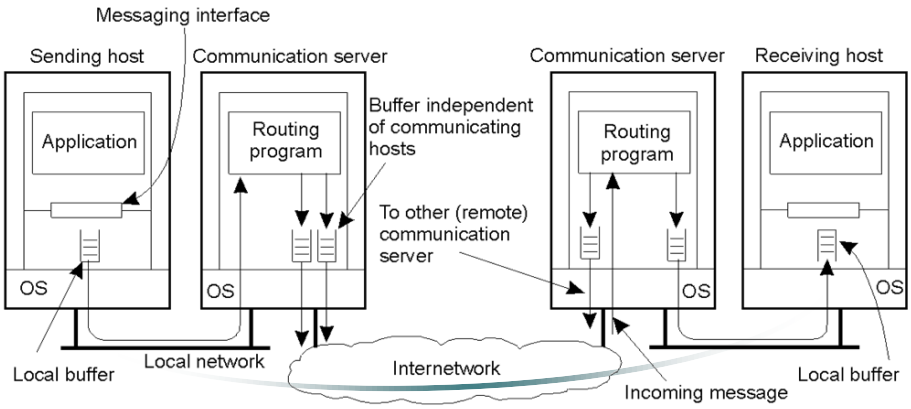
I.
II. Các kỹ thuật truyền thông
1. Mở đầu
2. Gọi thủ tục từ xa
3. **Truyền thông hướng thông điệp**

Message-Oriented Communication

- RPC: luôn có câu trả lời cho lời gọi
- Nếu không có câu trả lời → message-oriented communication (MOC)
  - Vậy ứng dụng có lấy được câu trả lời hay không?
  - Nếu không, điều gì xảy ra nếu ứng dụng cần xác nhận về 1 sự kiện đã xảy ra? Trong trường hợp này có dùng được MOC hay không?

Tính bền vững (persistency) và tính đồng bộ (synchronization)

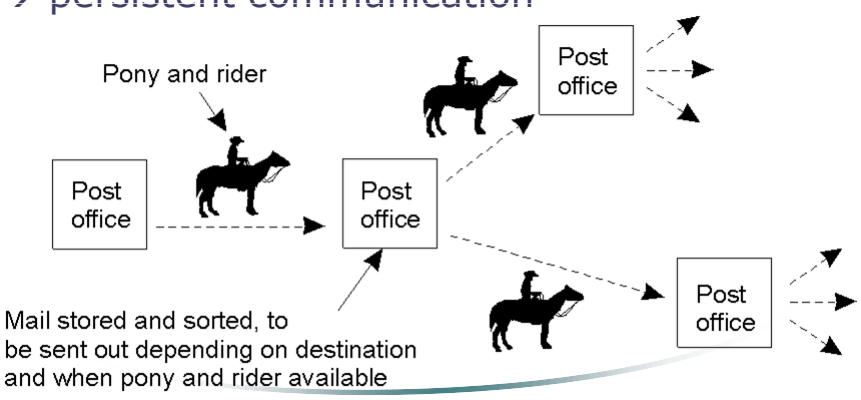
- Tổ chức của 1 hệ truyền thông trong đó các nút được nối mạng
- Cần quyết định lưu trữ tạm thời các thông điệp:
  - Chỉ lưu trữ trong thời gian đủ để xử lý và định tuyến
  - Lưu trữ nếu không gửi được
  - Lưu bao lâu?



Ví dụ: cơ chế store-and-forward

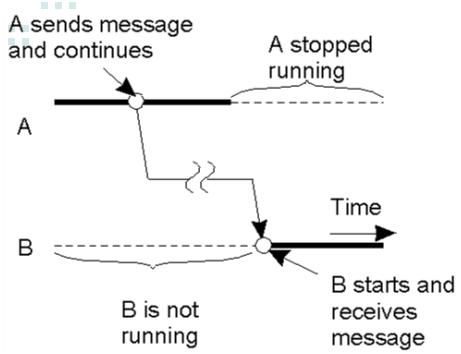
- Xử lý thư không giao được trong ngày của hãng Pony Express

→ persistent communication

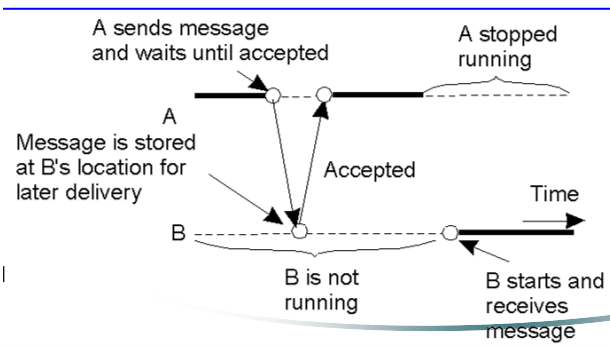


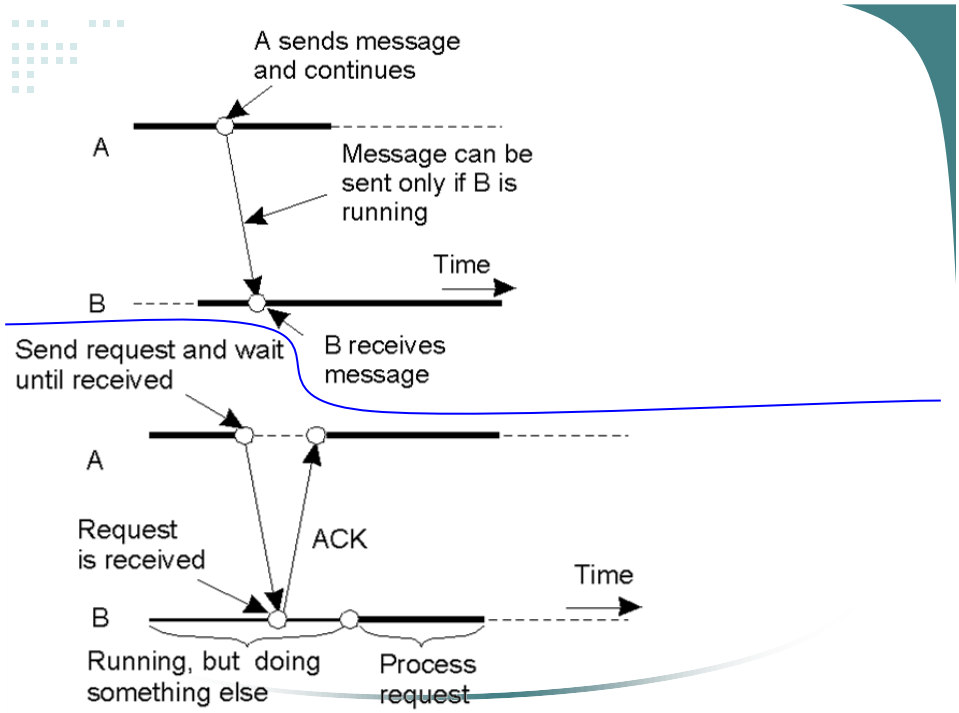
Phân loại các cách thức truyền thông

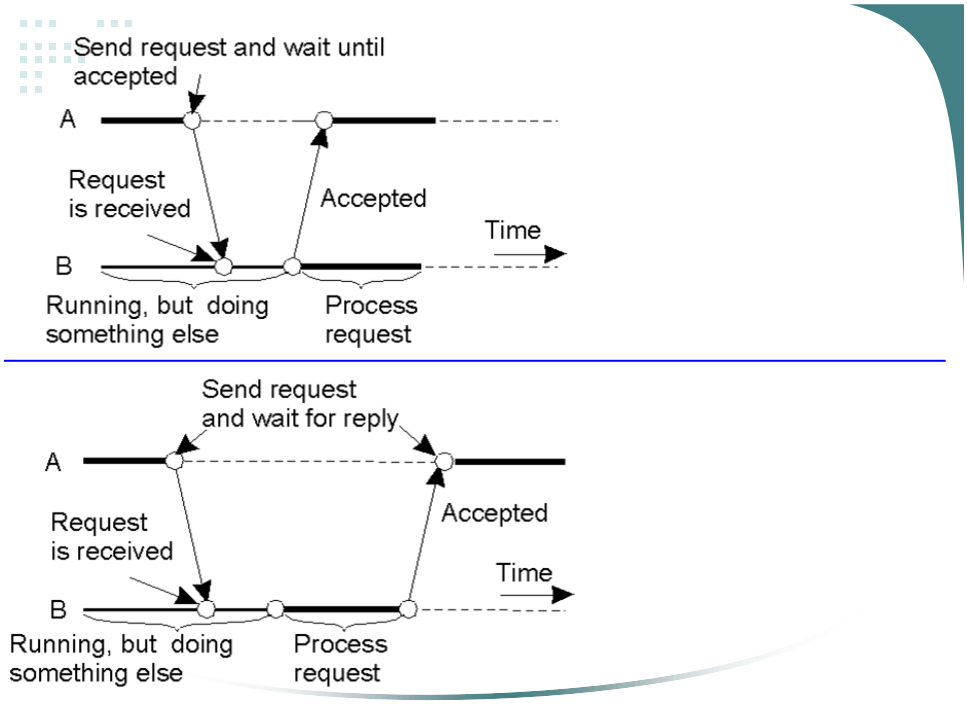
- Truyền thông nhất thời (Transient communication):
  - Bên gửi và bên nhận phải ở trạng thái hoạt động
  - Hệ truyền thông chỉ lưu trữ thông điệp khi nhận và khi gửi.
  - Nếu hệ truyền thông không chuyển giao được thì loại bỏ thông điệp
- Truyền thông bền vững (Persistent communication):
  - Bên gửi có thể ngừng hoạt động khi đã gửi xong
  - Bên nhận không cần ở trạng thái hoạt động khi thông điệp được chuyển giao
  - Hệ truyền thông giữ lại thông điệp không chuyển giao được trong bộ đệm
  - → Giữ trong bao lâu ? đến khi chuyển giao xong
- Truyền thông đồng bộ (Synchronous communication): Bên gửi đợi phản hồi đã nhận được thông điệp của bên nhận trước khi tiếp tục công việc
- Truyền thông không đồng bộ (Asynchronous communication): bên gửi tiếp tục công việc mà không đợi phản hồi từ bên nhận



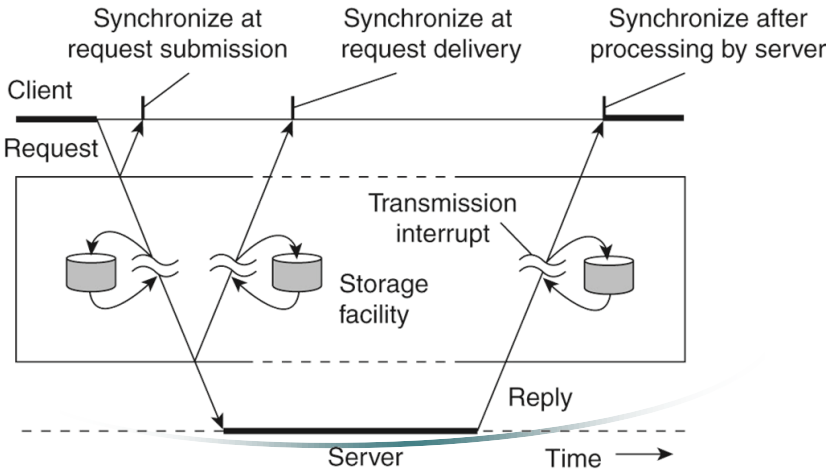
Đây là truyền thông loại gì?




A sends message
and continues
A
Message can be
sent only if B is
running
Time
B
B receives
message
Send request and wait
until received
A
Request
is received
ACK
Time
B
Running, but doing
something else
Process
request


Send request and wait until
accepted
A
Request
is received
Accepted
Time
B
Running, but doing
something else
Process
request
Send request
and wait for reply
A
Request
is received
Accepted
Time
B
Running, but doing
something else
Process
request

Truyền thông với middleware



4. TRUYỀN THÔNG HƯỚNG DÒNG (STREAM – ORIENTED COMMUNICATION)

GIANG VÝ

I. Các thuật toán
II. Các kỹ thuật truyền thông
1. Mở đầu
2. Gọi thủ tục từ xa
3. Truyền thông hướng thông điệp
4. **Truyền thông hướng dòng**
5. Truyền thông đa điểm

a. Đặt vấn đề

- Khi trao đổi các đơn vị thông tin nguyên vẹn
  - Không quan tâm đến yếu tố thời gian
- Với 1 số loại thông tin, thời gian là thiết yếu
  - Ví dụ phim ảnh, âm thanh.
- 2 loại media:
  - Biểu diễn liên tục (continuous media): quan hệ thời gian là chủ yếu.
  - Biểu diễn rời rạc (discrete media): quan hệ thời gian không là chủ yếu

GIANG VÝ

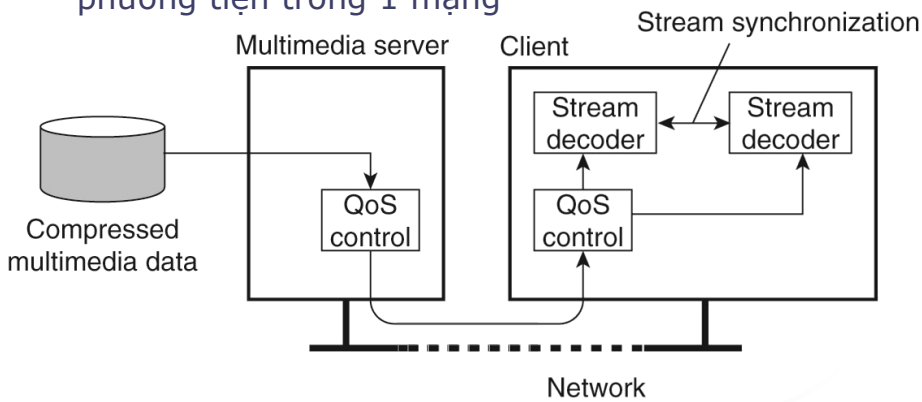
I. Các thuật toán
II. Các kỹ thuật truyền thông
1. Mở đầu
2. Gọi thủ tục từ xa
3. Truyền thông hướng thông điệp
4. **Truyền thông hướng dòng**
5. Truyền thông đa điểm

a. Đặt vấn đề

- Luồng dữ liệu (Data stream): dãy các đơn vị dữ liệu
  - Có thể liên tục hoặc rời rạc
- Kiểu truyền (Transmission modes):
  - Không đồng bộ (Asynchronous): không có ràng buộc về thời gian, thường áp dụng cho các luồng dữ liệu rời rạc.
  - Đồng bộ: thời gian trễ giữa bên gửi và bên nhận là lớn nhất
  - Đồng bộ cách ly (Isochronous): xác định khoảng thời gian truyền lớn nhất và nhỏ nhất (bounded jitter).
    - Tại sao cần biết thời gian truyền nhỏ nhất ?
- Đơn giản và phức tạp:
  - Đơn giản: 1 dãy các dữ liệu
  - Phức tạp: 1 số các luồng dữ liệu liên quan, gọi là luồng dữ liệu con

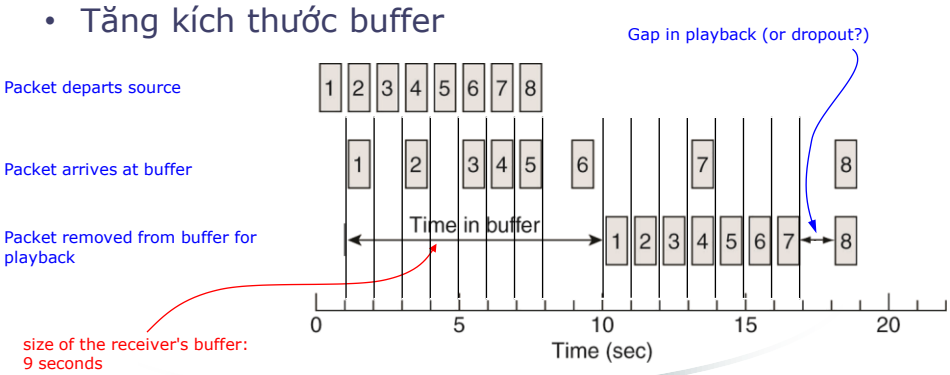
Kiến trúc

- Cho phép truyền theo luồng các dữ liệu đa phương tiện trong 1 mạng



Làm thế nào để nâng cao chất lượng dịch vụ Internet ?

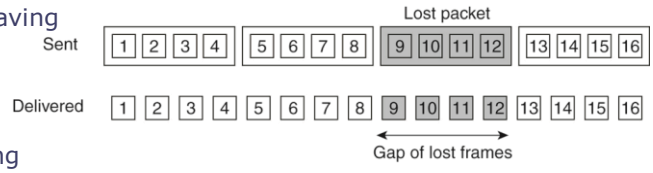
- Chất lượng dịch vụ
  - Thời gian
  - Các yêu cầu ngoài chức năng
- Tăng kích thước buffer



Làm thế nào để nâng cao chất lượng dịch vụ Internet ?

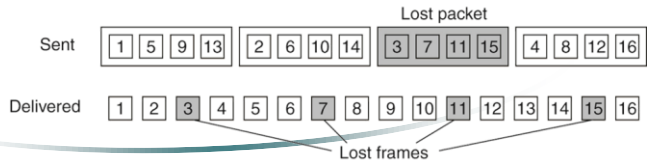
- Sửa chữa lỗi gửi chuyển tiếp (Forward error correction)
  - 1 gói tin có thể chứa 1 vài audio sample. Nếu gói tin bị mất → bị mất 1 đoạn lớn
  - Nếu mỗi audio sample bị cắt bớt 0.1s, hay mỗi giây cắt bớt 10% của 1 audio sample, điều gì xảy ra?

- Without interleaving



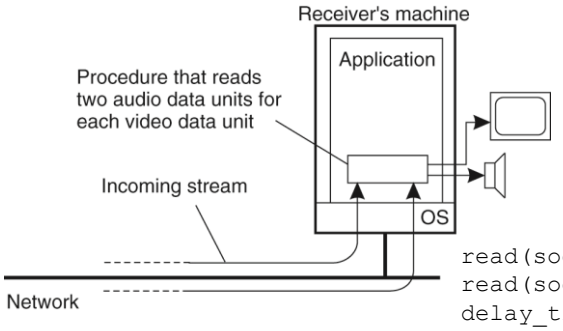
- With interleaving

Nhược điểm của interleaving?



Các kỹ thuật đồng bộ

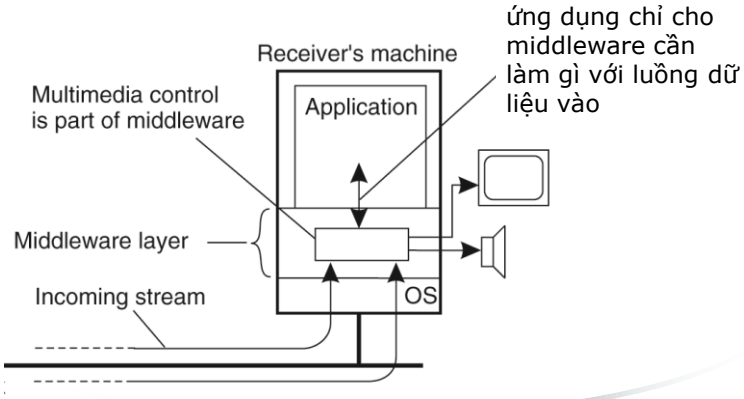
- Nếu các luồng phức tạp, cần đồng bộ thời gian giữa các luồng con (ví dụ: luồng âm thanh và luồng hình ảnh)
- Do ứng dụng thực hiện




```
read(sock_audio, img);
read(sock_audio, audio_block);
delay_till_next_frame();
write(video_hw, img);
write(audio_hw, audio_block);
```

Các kỹ thuật đồng bộ

- Thực hiện bởi middleware, điều khiển bởi ứng dụng qua giao diện bậc cao



GIANG

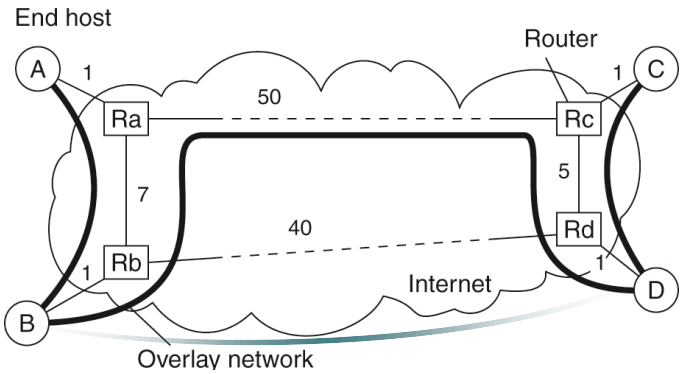
I. Các thuật toán
II. Các kỹ thuật truyền thông
1. Mở đầu
2. Gọi thủ tục từ xa
3. Truyền thông hướng thông điệp
4. Truyền thông hướng dòng
5. **Truyền thông đa điểm**

5. Truyền thông đa điểm (Multicast Communication)

- Truyền thông đa điểm, còn gọi là truyền thông theo nhóm, thường hoạt động ở tầng IP
  - Khó cấu hình, cần hỗ trợ của admin, hợp đồng, v.v.
- Truyền thông đa điểm ở mức ứng dụng cần khắc phục các vấn đề trên
- Giả sử tiến trình S muốn gửi 1 thông điệp đến N bên nhận (R_1 – R_N). Làm thế nào?

Xây dựng mạng chồng (overlay)

- Các kỹ thuật overlays có thể kém hiệu quả
  - Multicasting from A will traverse <B, Rb>, <Ra, Rb>, <Rc, Rd>, and <D, Rd> twice.
  - It would have been better to make an overlay link between A and C instead of B and D.



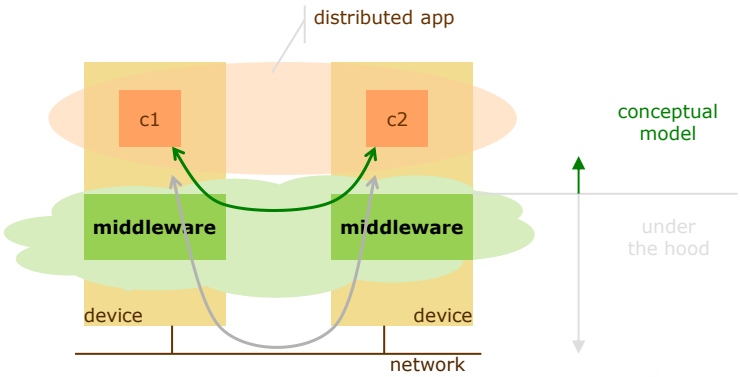
Kỹ thuật Gossiping

- "Tám" như thế nào
  - Nếu A định tám chuyện với B, B sẽ tám lại chuyện đó với C.
  - Nếu C chưa biết chuyện B định tám, B cảm thấy thỏa mãn, và muốn nói chuyện đó với người khác nữa
  - Nếu B tám chuyện với N người, và N người này đều biết chuyện đó rồi, B sẽ không muốn tám chuyện đó với ai khác nữa

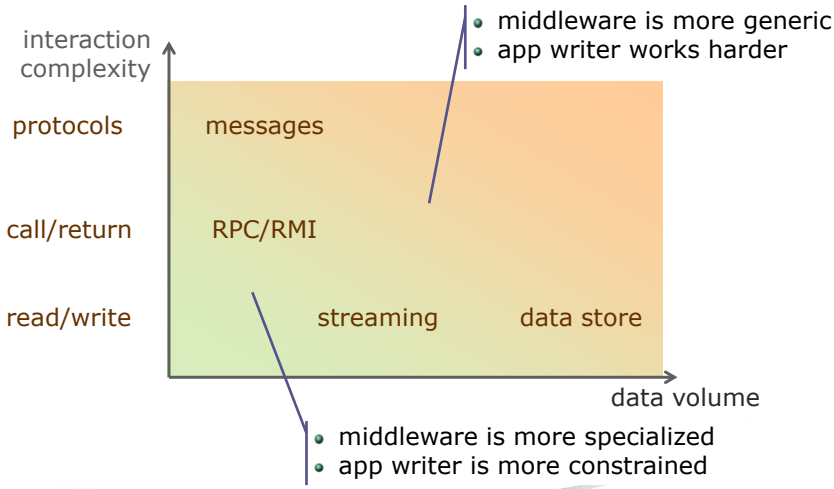
GIANG VÝ

Biểu diễn các mô hình truyền thông khái quát

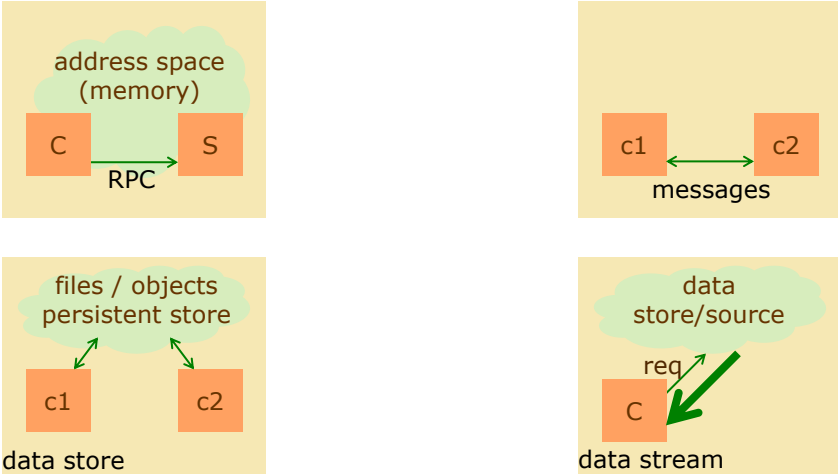
I. Các thuật toán
II. Các kỹ thuật truyền thông
1. Mở đầu
2. Gọi thủ tục từ xa
3. Truyền thông hướng thông điệp
4. Truyền thông hướng dòng
5. Truyền thông đa điểm
6. **Cài đặt các kỹ thuật nói trên**



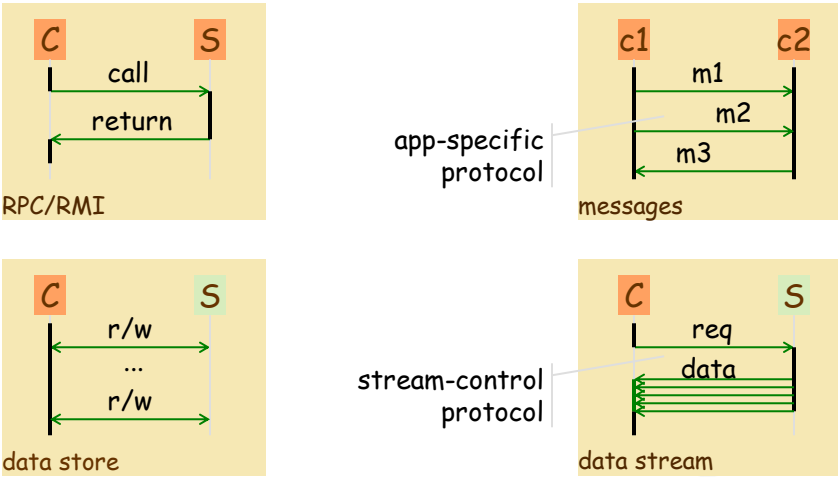
Các kiểu mô hình khái quát cho các bài toán truyền thông khác nhau



Giả thiết về vấn đề chia sẻ dữ liệu cho các kiểu mô hình truyền thông khác nhau



Giả thiết về vấn đề luồng điều khiển cho các kiểu mô hình truyền thông khác nhau



III. CÁC DỊCH VỤ HỖ TRỢ

GIANG

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)

a. Khái niệm

- Tên (name): dãy các bit, sử dụng tham chiếu đến 1 thực thể
  - Đặc tính cần có:
    - Định danh thực thể: Dễ nhớ
    - Tham chiếu đến vị trí của thực thể: Độc lập với vị trí
- Hệ tên (naming system): dùng để phân giải tên (name resolution), tức là tìm kiếm 1 thực thể trong HPT theo tên
  - Ví dụ: file systems, ZIP code, DNS

→ Cần xác định ánh xạ giữa tên và thực thể nó tham chiếu đến
  - Cho 1 tên và 1 thực thể, liệu có thể sử dụng thực thể được ngay hay không? (Tôi biết tên bạn, tôi có thể yêu cầu bạn mọi việc không ?)

GIANG

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)

a. Khái niệm

- Để thao tác với một thực thể, cần truy nhập vào thực thể đó thông qua 1 điểm truy nhập (access point).
- Tên của điểm truy nhập vào thực thể là địa chỉ của thực thể đó (address).
  - 1 thực thể có thể có nhiều điểm truy cập hay không ?
  - Điểm truy cập vào thực thể có thể thay đổi theo thời gian hay không?
- Tên là duy nhất hay không ? Vĩnh viễn hay không?
- Địa chỉ là duy nhất hay không ? Vĩnh viễn hay không?
- Bí danh (alias):
  - Hard alias: ví dụ 2 mã ZIP cho cùng một vùng
  - Soft alias: ví dụ forwarding address.

Ví dụ

- IDs là một kiểu tên. Nó cần có đặc tính gì?
- Tên bạn ? Địa chỉ của bạn ? ID của bạn ?
- Có thể dùng thay thế lẫn nhau được không ?

GIANG VÝ

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)

a. Khái niệm

- Không gian tên (name space):
  - Phương thức tổ chức tên trong một hệ thống cụ thể
  - Định nghĩa tập các tên có thể đặt cho các thành phần của hệ thống đó
- Ví dụ?
- Làm thế nào để mở rộng/ trộn lẫn không gian tên?

GIANG VÝ

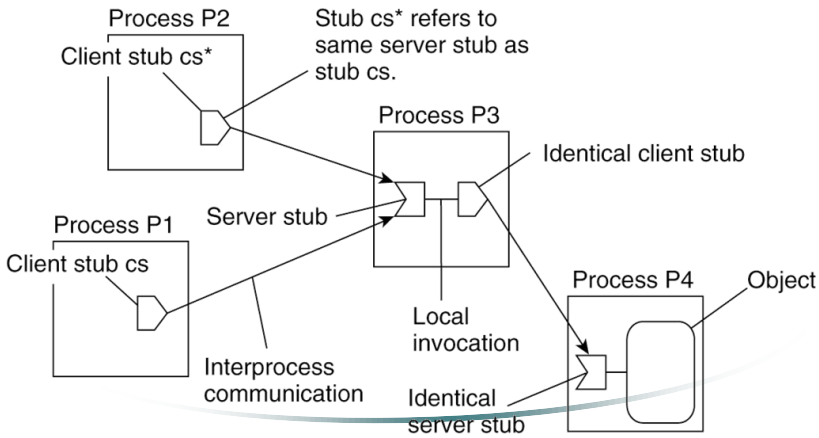
III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)

b. Một số kỹ thuật đặt tên tiêu biểu

- Flat Naming: định vị thực thể theo idenfier là các xâu ngẫu nhiên
  - Dựa trên vị trí (Home-based approach): Mobile IP
  - Bảng băm phân tán (DHT): Chord
- Structured Naming: định vị thực thể theo cách con người thường gọi tên thực thể
  - Human-friendly names: tên có cấu trúc, gồm nhiều phần
  - Ví dụ:
- Attribute-based Naming: tìm kiếm thực thể theo các thuộc tính của thực thể đó
  - Thực thể được miêu tả dưới dạng các cặp (thuộc tính, giá trị)
  - Ví dụ: X.500, LDAP
- Bài tập về nhà: tìm hiểu các kỹ thuật nêu trên

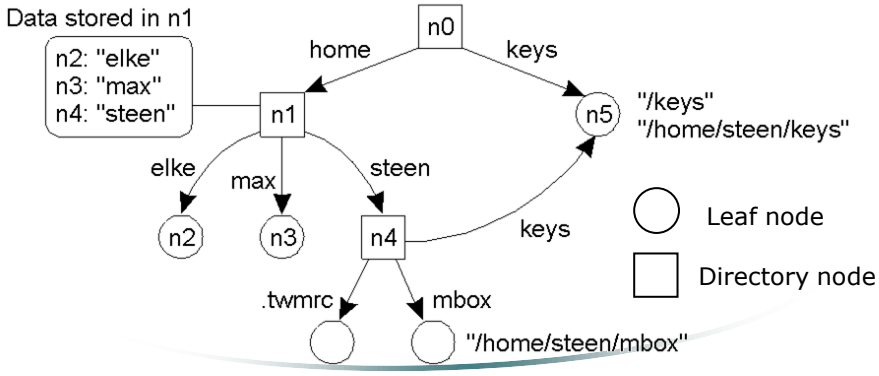
Ví dụ: flat naming
Forwarding Pointers in SSP

- Object originally in P3, then moved to P4. P1 passes object reference to P2.
  - Do we always need to go through the whole chain?



Ví dụ: structured naming
Naming Graph

- Path, local name, absolute name
- Should it be a tree, DAG, allow cycles?



Ví dụ: attribute-based Naming Directory Services

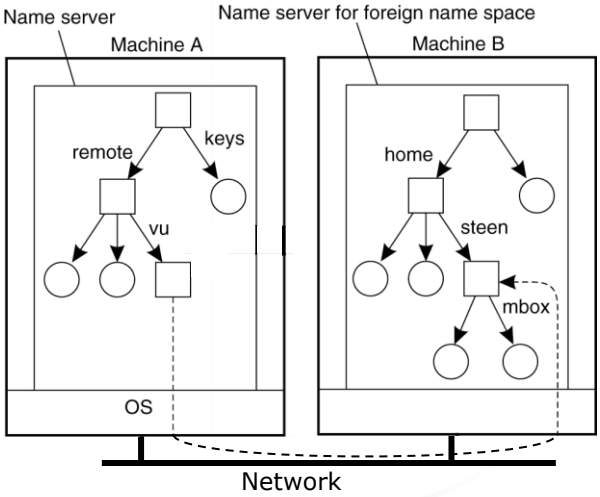
- Cho phép tìm 1 thứ dựa trên 1 loại miêu tả về thứ đó
- Resource Description Framework (RDF)
  - (Person, name, Alice): describes a resource Person whose name is Alice.

c. Kỹ thuật mở rộng không gian tên mounting

- Xét tập các không gian tên phân tán trên nhiều máy (mỗi máy có 1 không gian tên)
- Thông tin cần thiết để gắn một không gian tên vào không gian tên của 1 HPT:
  - Tên giao thức truy nhập
  - Tên server
  - Tên của mounting point: nút thư mục đặc biệt trong không gian tên của 1 HPT để nhảy sang không gian tên khác

Bài tập

- Mounting remote name spaces through a specific process protocol.
- How do you access steen's mbox from Machine A?



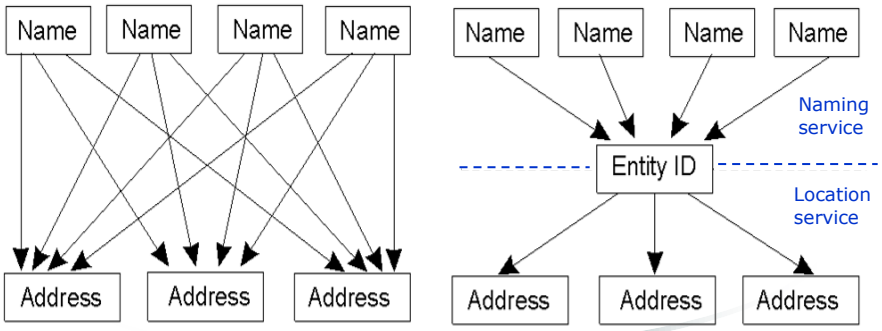
d. Naming vs. Locating

- Làm thế nào để chuyển từ tên sang địa chỉ?
- Làm thế nào để thay đổi từ ftp.it.hut.edu.vn sang ftp.soict.hut.edu.vn ?

- Giải pháp tốt hơn là phân chia thành 2 dịch vụ: naming service và location service.
  - Bao nhiêu mức đặt tên được sử dụng trên Internet?

Direct, single level mapping between names and addresses.

Two-level mapping using identities and separate naming and location service.



GIANG

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)
2. **Kết nối (Binding)**

a. Khái niệm

- Binding: Liên kết giữa một tên và 1 địa chỉ
  - "choose a lower-level-implementation for a higher-level semantic construct"

→ Liên kết trong quá trình phân giải tên

GIANG VÝ

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)
2. **Kết nối (Binding)**

b. Phân loại

- Static binding
  - Hard-coded
- Early binding
  - Tìm kiếm kết nối trước khi sử dụng
  - Giấu đi các kết nối đã được sử dụng trước đó
- Late binding
  - Tìm kiếm ngay trước khi sử dụng

GIANG VÝ

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)
2. **Kết nối (Binding)**

c. Binder trong RPC

- Vấn đề: làm thế nào 1 client định vị được 1 server
  - Dùng bindings
- Server
  - Khi khởi tạo server, tự giới thiệu giao diện
  - Gửi các thông tin sau đến binder: tên, số hiệu phiên bản đang hoạt động, ID duy nhất, các tín hiệu điều khiển (bao gồm địa chỉ)
- Client
  - Gửi thông điệp đến cho binder để lấy thông tin về giao diện của server
- Binder:
  - Kiểm tra xem server đã giới thiệu giao diện của nó hay chưa
  - Trả về các tín hiệu điều khiển và ID duy nhất của server cho client

GIANG

III. Các dịch vụ hỗ trợ
1. Đặt tên (Naming)
2. **Kết nối (Binding)**

3. Đồng bộ

- Synchronization: làm đúng việc cần làm vào đúng thời điểm
  - Đúng việc cần làm: giải thuật phân tán
  - Đúng thời gian:
    - Vấn đề: không có khái niệm về 1 đồng hồ chung cho tất cả các tiến trình
    - Giải pháp:
      - Trao đổi thời gian giữa các tiến trình khi gửi và nhận thông điệp giữa các tiến trình
      - Giải quyết các vấn đề đặt ra khi việc trao đổi thông tin giữa các tiến trình bị trễ
  - Bài tập về nhà: Đọc lại các giải thuật phân tán, tìm hiểu vấn đề đồng bộ về mặt thời gian khi thực hiện các GTPT đó

IV. MỘT SỐ KIẾN TRÚC TIÊU BIỂU

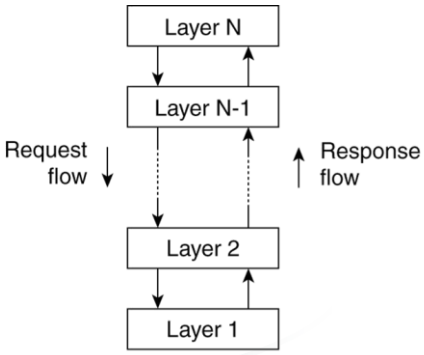
GIANG

IV. Một số kiến trúc tiêu biểu

1. **Các kiểu kiến trúc thường gặp**
2. Các kiến trúc tập trung
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

a. Kiến trúc phân tầng

- Các thành phần phần mềm tầng trên chỉ có thể gọi các thành phần phần mềm tầng dưới
  - Mềm dẻo: thêm chức năng mới mà không cần thay đổi các tầng liên quan
  - Tái sử dụng
  - Chia nhỏ vấn đề để giải quyết dễ dàng hơn



Ví dụ: đây có phải là kiến trúc phân tầng hay không?

- Tầng A ở trên tầng B nếu việc thay đổi các giao diện ở tầng A không đòi hỏi việc thay đổi mã nguồn tầng B.

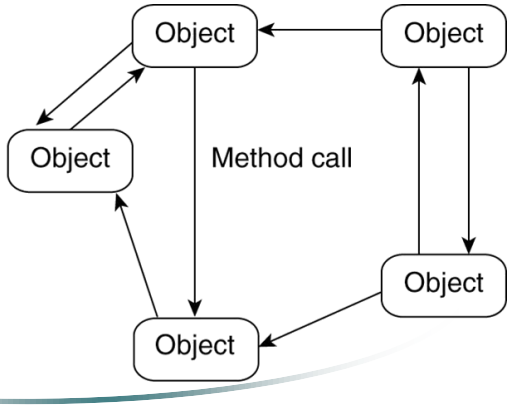
```
int forward_cmp(double x, double y) {
 return x > y;
}
int reverse_cmp(double x, double y) {
 return y > x;
}
void top_layer_func(…) {
 topo_sort(reverse_cmp, array, 10);
}

void topo_sort(
 int (*cmp)(double x, double y),
 double *array, int size) {
… }
```

IV. Một số kiến trúc tiêu biểu
1. **Các kiểu kiến trúc thường gặp**
2. Các kiến trúc tập trung
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

b. Kiến trúc dựa trên các đối tượng phân tán

- Đối tượng phân tán
- Việc thực hiện các lời gọi tương đối khó khăn.

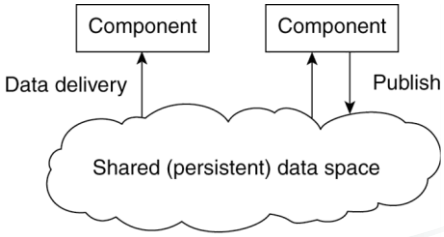


GIANG VÝ

IV. Một số kiến trúc tiêu biểu
1. **Các kiểu kiến trúc thường gặp**
2. Các kiến trúc tập trung
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

c. Kiến trúc tập trung vào dữ liệu

- Tương ứng với mô hình chia sẻ dữ liệu và/hoặc publish/subscribe
- Các tiến trình giao tiếp thông qua bộ nhớ dùng chung
- Ví dụ: WebDAV, Linda, tuple-space.

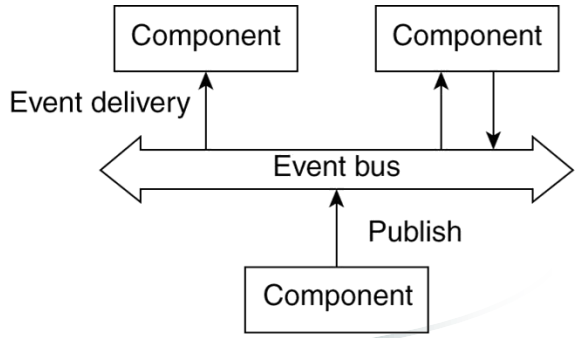


GIANG VÝ

IV. Một số kiến trúc tiêu biểu
1. **Các kiểu kiến trúc thường gặp**
2. Các kiến trúc tập trung
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

d. Kiến trúc dựa trên các sự kiện

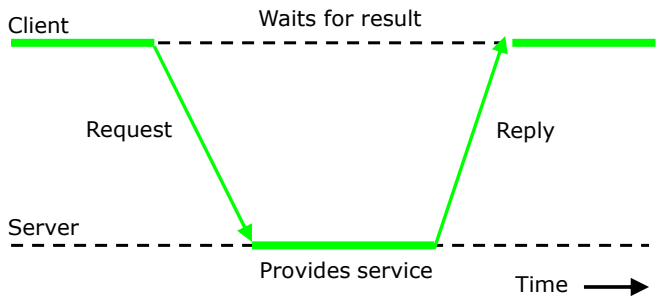
- Tương ứng với mô hình publish-subscribe áp dụng cho các sự kiện
- Đặc tính chính: phân tách các sự kiện theo không gian (vô danh – anonymous) và thời gian (không đồng bộ)



GIANG VÝ

IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. **Các kiến trúc tập trung**
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

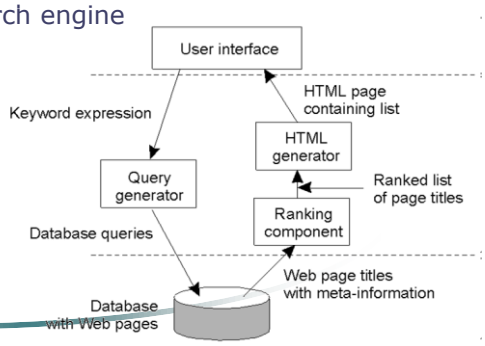
a. Client-Server



IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. **Các kiến trúc tập trung**
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

b. Các kiến trúc 3 tầng (3-Tier)

- Phân biệt được:
  - Giao diện người dùng
  - Logic nghiệp vụ (quy trình xử lý)
  - Dữ liệu (CSDL)
- Ví dụ các tầng này?
  - Internet search engine

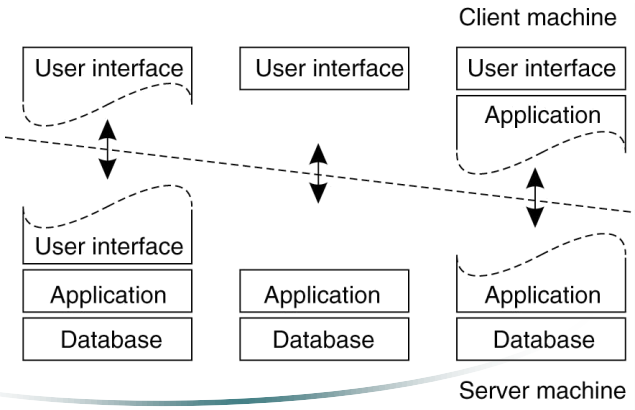


GIẢNG VỤ

IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. **Các kiến trúc tập trung**
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

b. Các kiến trúc 3 tầng (3-Tier)

- Kiến trúc 3 tầng vật lý có thể giống hoặc không giống kiến trúc 3 tầng logic, phụ thuộc vào các phân chia các phần chức năng chạy trên máy client và server

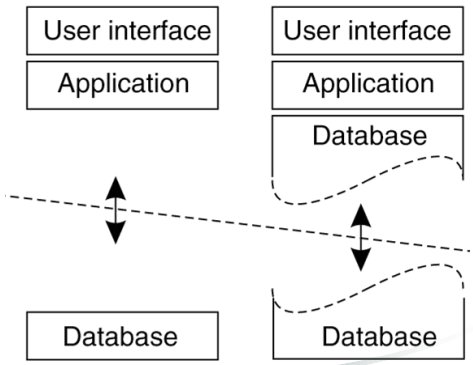


GIANG VÝ

IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. **Các kiến trúc tập trung**
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

b. Các kiến trúc 3 tầng (3-Tier)

- Kiến trúc 3 tầng vật lý có thể giống hoặc không giống kiến trúc 3 tầng logic, phụ thuộc vào các phân chia các phần chức năng chạy trên máy client và server

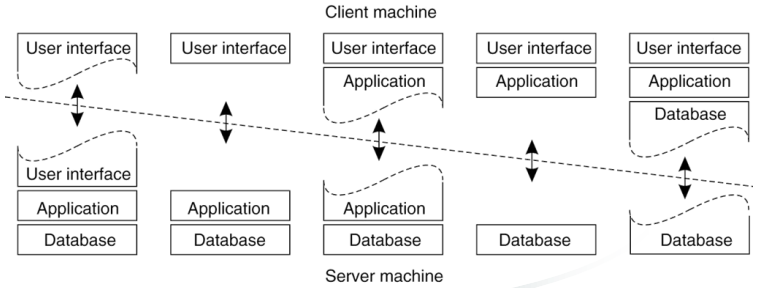


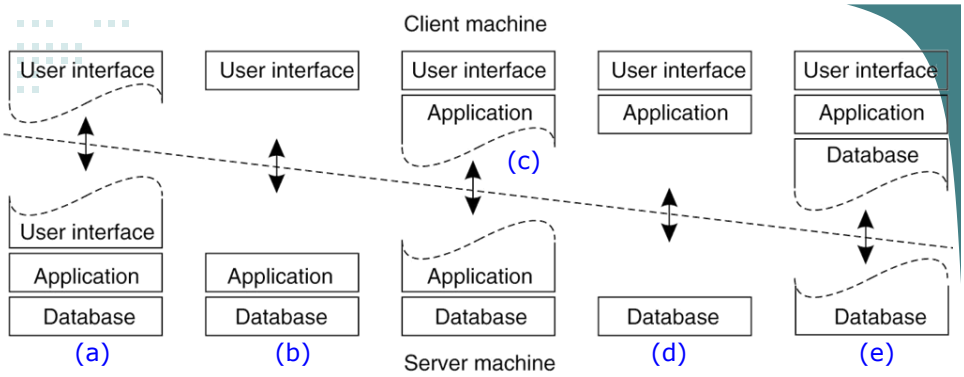
GIANG VÝ

IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. **Các kiến trúc tập trung**
3. Các kiến trúc không tập trung
4. Các kiến trúc lai

b. Các kiến trúc 3 tầng (3-Tier)

- Kiến trúc 3 tầng vật lý có thể giống hoặc không giống kiến trúc 3 tầng logic
- Phụ thuộc vào cách phân chia các phần chức năng chạy trên máy client và server



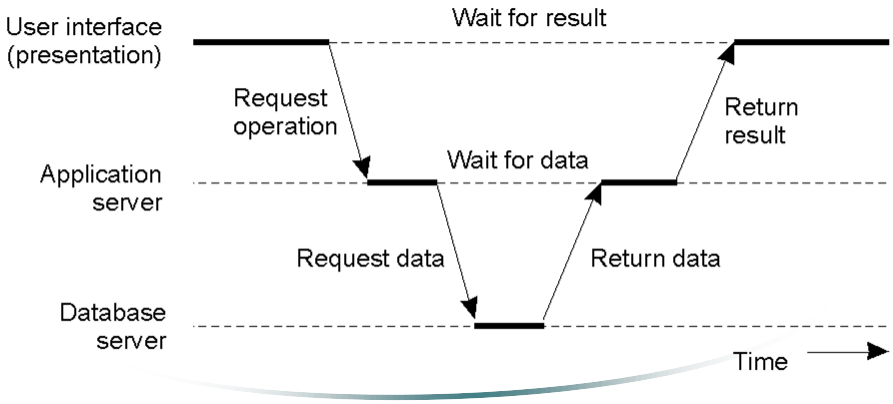


- Ví dụ:

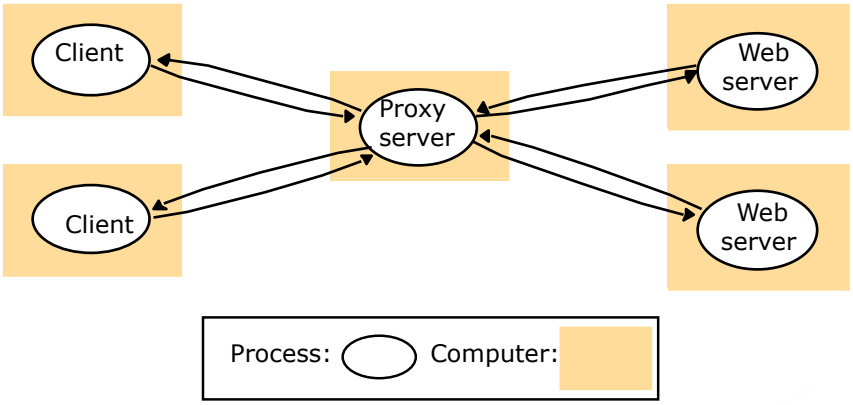
- Ưu nhược điểm của việc thực hiện 1 phần chức năng trên máy client?

Ví dụ kiến trúc 3 tầng vật lý

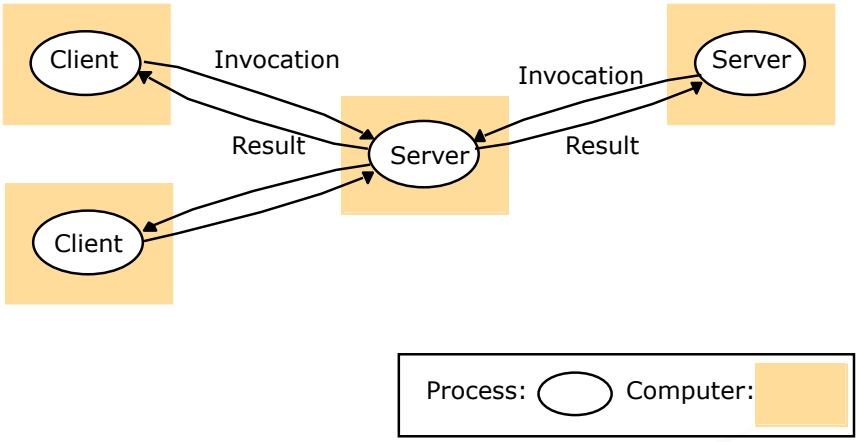
- Server hoạt động như client
  - Web server



Ví dụ:Web Proxy Server



Ví dụ: Gọi các server khác nhau



Ví dụ: Client và Server Header file

```
/* Definitions needed by clients and servers. */
#define MAX_PATH        255     /* maximum length of file name */
#define BUF_SIZE        1024    /* how much data to transfer at once */
#define FILE_SERVER     243     /* file server's network address */

/* Definitions of the allowed operations */
#define CREATE          1       /* create a new file */
#define READ            2       /* read data from a file and return it */
#define WRITE           3       /* write data to a file */
#define DELETE          4       /* delete an existing file */

/* Error codes. */
#define OK              0       /* operation performed correctly */
#define E_BAD_OPCODE    -1      /* unknown operation requested */
#define E_BAD_PARAM     -2      /* error in a parameter */
#define E_IO            -2      /* disk error or other I/O error */

/* Definition of the message format */
struct message {
  long source;          /* sender's identity */
  long dest;            /* receiver's identity */
  long opcode;          /* requested operation */
  long count;           /* number of bytes to transfer */
  long offset;          /* position in file to start I/O */
  long result;          /* result of the operation */
  char name[MAX_PATH];  /* name of file being operated on */
  char data[BUF_SIZE];  /* data to be read or written */
};
```

Ví dụ: mã nguồn phía server

```
#include <header.h>
void main(void){
    struct message m1, m2;      /* incoming and outgoing messages */
    int r;                      /* result code */

    while (TRUE) {                  /* server runs forever */
      receive(FILE_SERVER, &m1);    /* block waiting for a message */
      switch(m1.opcode){            /* dispatch on type of request */
          case CREATE:  r = do_create(&m1, &m2); break;
          case READ:    r = do_read(&m1, &m2); break;
          case WRITE:   r = do_write(&m1, &m2); break;
          case DELETE:  r = do_delete(&m1, &m2); break;
          default:      r = E_BAD_OPCODE;
      }
      m2.result = r;                /* return result to client */
      send(m1.source, &m2);         /* send reply */
    }
}
```

Ví dụ: mã nguồn client copying file

```
#include <header.h>
int copy(char *src, char *dst){   /* procedure to copy file using the server */
  struct message m1;              /* message buffer */
  long position;                  /* current file position */
  long client = 110;              /* client's address */

  initialize();                   /* prepare for execution */
  position = 0;
  do{
      m1.opcode = READ;           /* operation is a read */
      m1.offset = position;       /* current position in the file */
      m1.count = BUF_SIZE;        /* how many bytes to read */
      strcpy(&m1.name, src);      /* copy name of file to be read to message */
      send(FILESERVER, &m1);      /* send the message to the file server */
      receive(client, &m1);       /* block waiting for the reply */

      /* Write the data just received to the destination file. */
      m1.opcode = WRITE;          /* operation is a write */
      m1.offset = position;       /* current position in the file */
      m1.count = m1.result;       /* how many bytes to write */
      strcpy(&m1.name, dst);      /* copy name of file to be written to buf */
      send(FILE_SERVER, &m1);     /* send the message to the file server */
      receive(client, &m1);       /* block waiting for reply */
      position += m1.result;      /* m1.result is number of bytes written */
  }while(m1.result > 0);          /* iterate until done */
  return (m1.result >= 0 ? OK : m1.result);/* return OK or error code */
}
```

GIANG

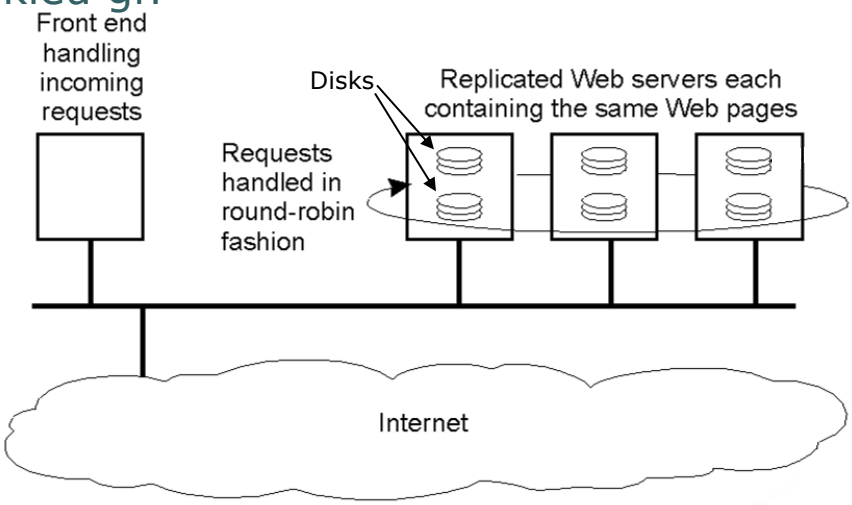
IV. Một số kiến trúc tiêu biểu

1. Các kiểu kiến trúc thường gặp
2. Các kiến trúc tập trung
3. **Các kiến trúc không tập trung**
4. Các kiến trúc lai

Phân bố dọc và phân bố ngang

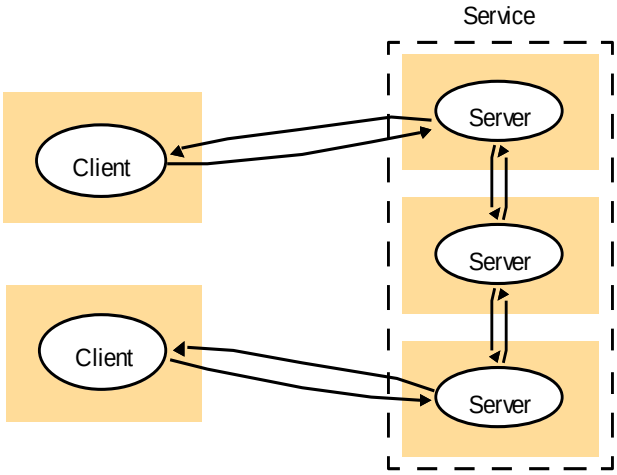
- Vertical distribution: phân chia ứng dụng theo chiều dọc
- Horizontal distribution: sao lưu, cụm máy tính

Ví dụ: dịch vụ web này được phân bổ kiểu gì?



horizontal distribution

Server phân bố dọc có thể giao tiếp với nhau

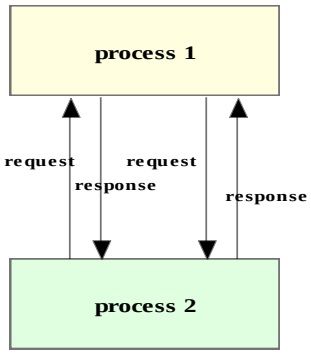


GIANG

IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. Các kiến trúc tập trung
3. **Các kiến trúc không tập trung**
4. Các kiến trúc lai

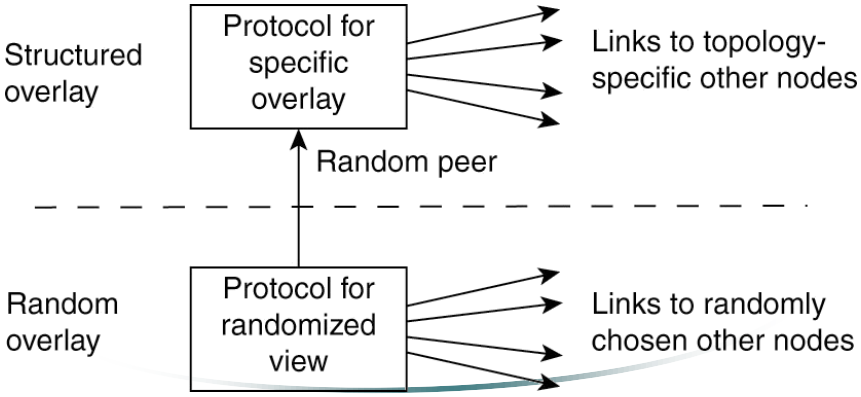
a. Peer-to-Peer

- Cho phép các máy tính trong mạng trao đổi trực tiếp tài nguyên và dịch vụ máy tính: thông tin, quy trình xử lý, lưu trữ bộ nhớ đệm, hay lưu trữ trong đĩa
- Các máy tính có vai trò, quyền lợi và trách nhiệm ngang nhau
- Phân loại:
  - Unstructured P2P: xây dựng mạng chồng với các nút ngẫu nhiên
  - Structured P2P: xây dựng mạng chồng với các nút được lựa chọn theo kỹ thuật DHT



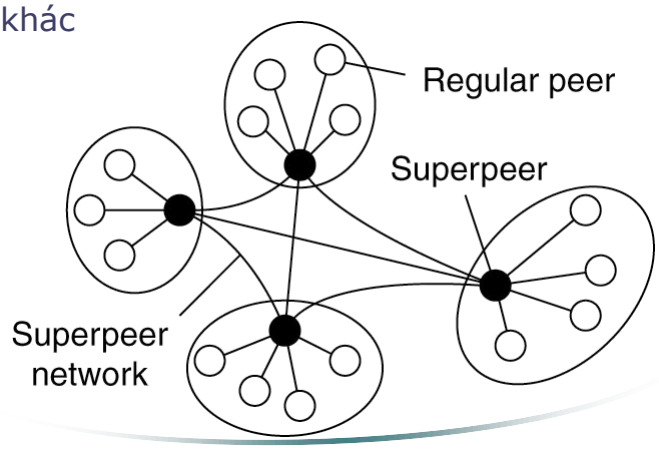
Hybrid Approaches

- Tầng dưới hoạt động độc lập



Superpeers

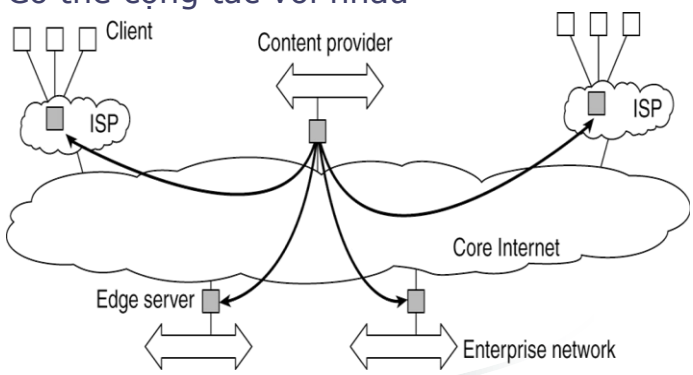
- Superpeers: các nút đặc biệt, đóng vai trò brokers hay giữ các tham số cho phép định vị các nút khác



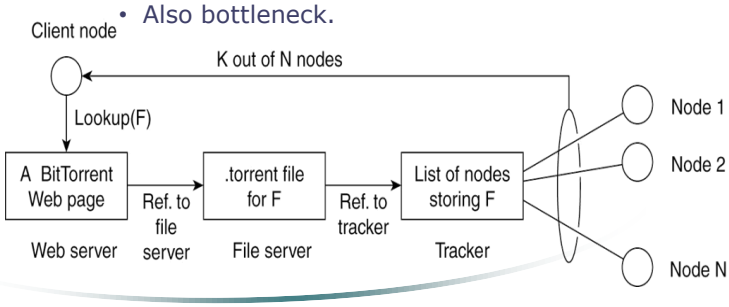
IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. Các kiến trúc tập trung
3. Các kiến trúc không tập trung
4. **Các kiến trúc lai**

Kiến trúc server bắc cầu (Edge Server Architecture)

- Các server bắc cầu kết nối người dùng vào mạng Internet.
- Có thể cộng tác với nhau



IV. Một số kiến trúc tiêu biểu
1. Các kiểu kiến trúc thường gặp
2. Các kiến trúc tập trung
3. Các kiến trúc không tập trung
4. **Các kiến trúc lai**

Các hệ cộng tác phân tán

- BitTorrent
  - Làm thế nào thuyết phục người dùng vừa thu thập thông tin, vừa phân phát thông tin?
    - Exchange incentive system
  - Trackers được xử lý tập trung
    - Also bottleneck.