

25 YEARS ANNIVERSARY

SOKT

The graphic consists of the number "25" in large, bold, white font. A curved banner arches over the top of the "2" containing the text "YEARS ANNIVERSARY". Below the "25" is the acronym "SOKT" in a large, bold, white, sans-serif font.

**ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Chương 1: Tổng quan và kiến trúc Hệ Phân Tán

Nội dung

1. Định nghĩa
2. Đặc điểm của hệ phân tán
3. Kiến trúc phần mềm và Kiến trúc hệ thống
4. Middleware trong Hệ phân tán



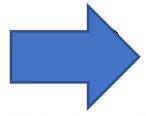
ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

1. Định nghĩa

- 1.1. Lịch sử phát triển
- 1.2. Các định nghĩa
- 1.3. Ví dụ

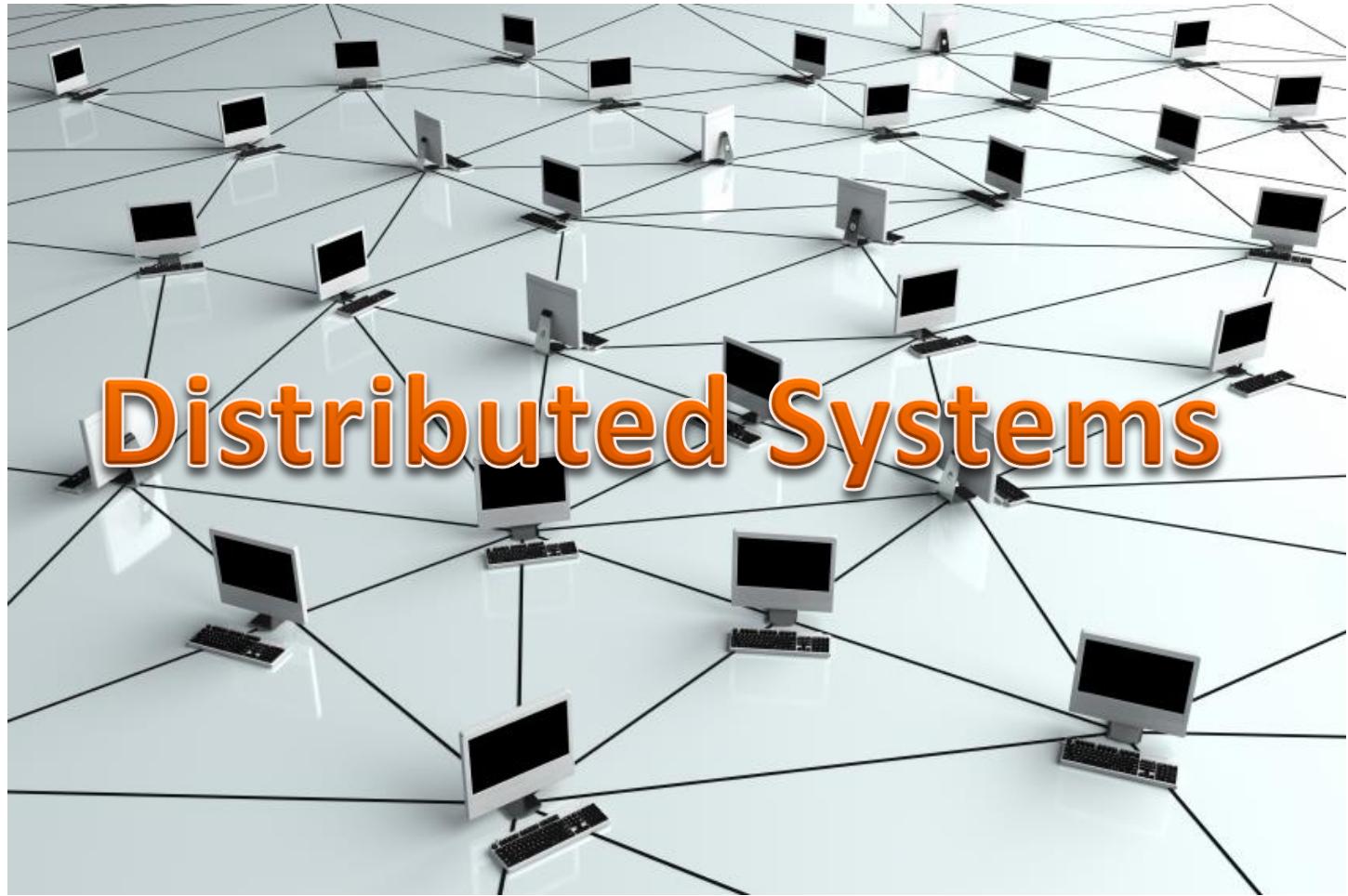
1.1. Lịch sử phát triển của các hệ thống máy tính

- Lịch sử phát triển các hệ thống máy tính
 - Thế hệ máy tính thứ nhất (1945 – 1956)
 - Bóng đèn chân không
 - ENIAC (Electronic Numerical Integrator And Computer)
 - Thế hệ thứ hai (1958-1964)
 - Transistor
 - Thế hệ thứ ba (1965-1971)
 - IC: Integrated Circuit
 - Thế hệ thứ tư (1972-nay)
 - VLSI: Very Large Scale Integration
- Lịch sử phát triển Mạng máy tính



Thay đổi về cách thức sử dụng máy tính

Các hệ thống phân tán



1.2. Định nghĩa

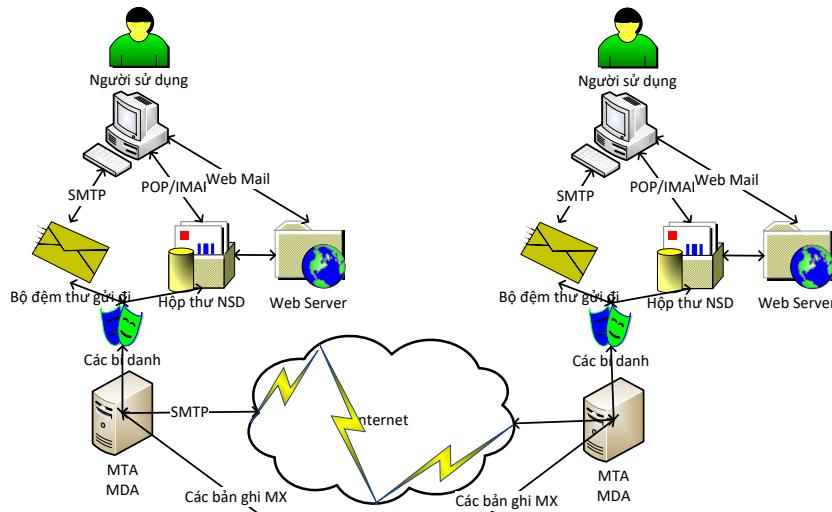
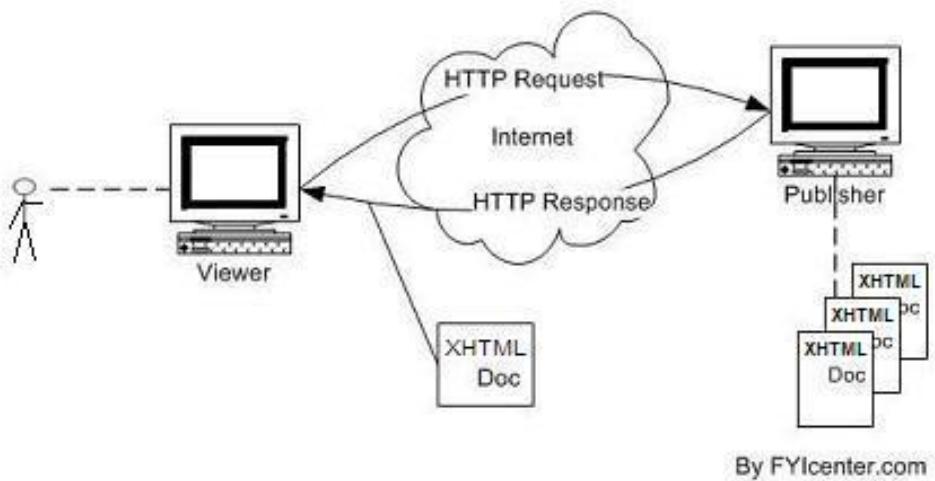
- Các máy tính độc lập
 - Không phụ thuộc lẫn nhau, có thể là các máy tính có kiến trúc khác nhau, có thể là các máy tính có phần mềm hệ thống khác nhau
- Kết nối lẫn nhau
 - Bằng mạng máy tính. Các phần mềm trên các máy tính khác nhau có khả năng phối hợp. Chia sẻ tài nguyên.
- Thực hiện một nhiệm vụ chung
- Cung cấp dịch vụ một cách thống nhất
 - Thống nhất về giao diện, cách thức truy cập dịch vụ → mức độ thống nhất
- NSD không cần phải quan tâm tới các chi tiết của hệ thống
 - *A collection of independent connected computers that provides services to its users as a single coherent system.*
[Tanenbaum 2006]

Distributed vs. Ubiquitous Systems

- Networked computer system: appears as many machines
- Distributed computer system: appears as single system
- Ubiquitous system: appears as no computer system

1.3. Ví dụ về hệ phân tán

- Hệ thống WWW
- Hệ thống Email
- V.v...





ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

2. Đặc điểm của hệ phân tán

- 2.1. Chia sẻ tài nguyên
- 2.2. Tính trong suốt
- 2.3. Tính mở
- 2.4. Tính co giãn (scalability)

Các đặc trưng của một hệ thống phân tán

- Chia sẻ tài nguyên
- Tính mở
- Tính trong suốt
- Tính co giãn

2.1. Chia sẻ tài nguyên

- Kết nối tài nguyên
- Giảm chi phí
- Tăng tính sẵn sàng
- Hỗ trợ làm việc nhóm
- Tăng rủi ro về an toàn thông tin

2.2. Tính trong suốt (transparency)

- Hệ thống là duy nhất với NSD
 - Giao diện giống nhau
 - Cách thức truy cập giống nhau
- Trong suốt về qui mô và vị trí
- **Che giấu tính phân tán của hệ phân tán**
- Các loại trong suốt (slide sau)
- Mức độ trong suốt:
 - **Cân bằng giữa hiệu năng và độ trong suốt**

Các loại trong suốt

Loại trong suốt	Mô tả
Truy cập	Che giấu sự khác nhau trong biểu diễn dữ liệu và cách thức truy cập tài nguyên.
Địa điểm	Che giấu vị trí của tài nguyên
Di trú	Che giấu việc tài nguyên chuyển đến địa điểm khác
Chuyển địa điểm	Che giấu việc tài nguyên chuyển đến địa điểm khác trong khi đang được sử dụng
Sao lưu	Che giấu việc dữ liệu được cung cấp từ nhiều bản sao khác nhau
Tương tranh	Che giấu việc tài nguyên được truy cập đồng thời bởi nhiều NSD
Thứ lỗi	Che giấu lỗi và quá trình phục hồi của tài nguyên
Bền vững	Che giấu việc tài nguyên/dữ liệu được lưu trữ bền vững (disk) hoặc không (RAM)

2.3. Tính mở

- Cho phép các thành phần có thể được sản xuất bởi các NSX khác nhau.
- Hệ phân tán mở cung cấp các dịch vụ theo các đặc tả về cú pháp và ngữ nghĩa của các dịch vụ, gọi là **giao diện**
- Thường được mô tả bằng **IDL**
- Tính đầy đủ của đặc tả
 - **Quá chi tiết:** phụ thuộc vào cài đặt cụ thể của dịch vụ
 - **Không đủ chi tiết:** Khi cài đặt phải bổ sung thêm: phụ thuộc vào cài đặt cụ thể của dịch vụ

Tính mở (2)

- Khả năng phối hợp (interoperability)
- Tính khả chuyển (portability)
- Tính mềm dẻo + mở rộng được (flexibility, extensibility)
- Thực hiện: tách biệt chính sách và cơ chế

2.4. Tính co giãn

- Qui mô:
 - số lượng NSD và tài nguyên thay đổi
- Không gian địa lý
 - Qui mô vùng địa lý có tài nguyên và NSD thay đổi
- Tổ chức
 - Qui mô tổ chức thay đổi → tổ chức hệ thống thành các domain.

Co giãn theo số lượng

- Mô hình tập trung
 - Dịch vụ: cổ chai
 - Dữ liệu: lưu trữ, xử lý
 - Giải thuật: thông tin vào ra, xử lý
- Mô hình không tập trung
 - Phức tạp, vđ về bảo mật và riêng tư
 - Quyết định cục bộ
 - Không có thông tin chung
 - Không phát hiện được lỗi

Cơ giàn theo không gian địa lý

- Gần: **mạng cục bộ**
 - quảng bá, tốc độ cao, tin cậy, độ trễ cố định)
- Xa: mạng diện rộng
 - Điểm điểm, tốc độ thấp, không tin cậy, độ trễ thay đổi
- Khác nhau
 - Tốc độ truyền tin, độ trễ,
 - Đồng bộ/không đồng bộ
 - Các thao tác quảng bá
- Chủ yếu đảm bảo trao đổi thông tin trên mạng diện rộng như với mạng cục bộ



ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

3. Kiến trúc phần mềm và Kiến trúc hệ thống

3.1. Kiến trúc

- Xem xét tổ chức của một Hệ Phân Tán → tách biệt giữa *tổ chức logic* và *thực thi vật lý*.
- Tổ chức logic: các thành phần phần mềm, cách thức kết nối, kiểu dữ liệu trao đổi → **Kiến trúc phần mềm**
- Thực thi vật lý: cách thức xếp đặt/cài đặt các thành phần phần mềm lên các thiết bị vật lý → **Kiến trúc hệ thống**

3.2. Các kiểu kiến trúc phần mềm thường dùng trong hệ phân tán

- Kiến trúc phân tầng
- Kiến trúc hướng đối tượng
- Kiến trúc hướng dữ liệu
- Kiến trúc hướng sự kiện
- Kiến trúc Microservices

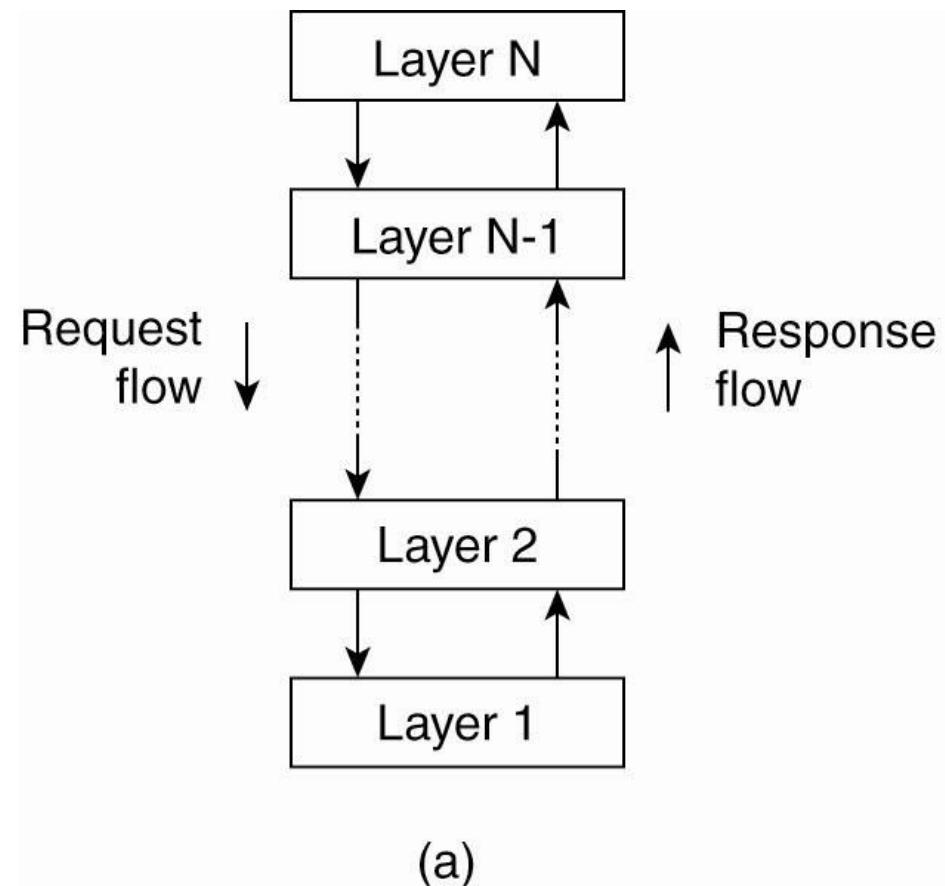
3.2.1. Kiến trúc phân tầng

- Chức năng trên hệ thống được phân rã thành các chức năng con
- Các chức năng con được thực hiện bởi các mô đun phần mềm – các thực thể phần mềm trên các hệ thống khác nhau tương tác với nhau
- Các mô đun phần mềm khác nhau trên cùng hệ thống phối hợp và tương tác với nhau để thực hiện chức năng chung
- Để đơn giản hệ thống cần giảm thiểu liên kết giữa các mô đun: kiến trúc phân tầng

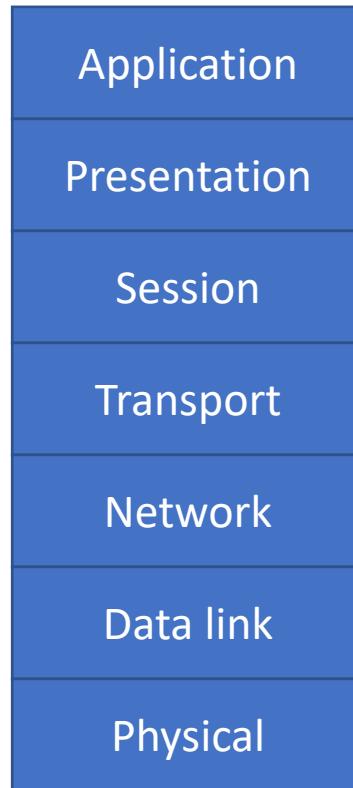
Kiến trúc phân tầng

Đặc điểm

- Trong suốt giữa các tầng
- Các tầng liền kề trao đổi thông tin:
 - Luồng yêu cầu
 - Luồng trả lời



Các mô hình phân tầng thường gặp

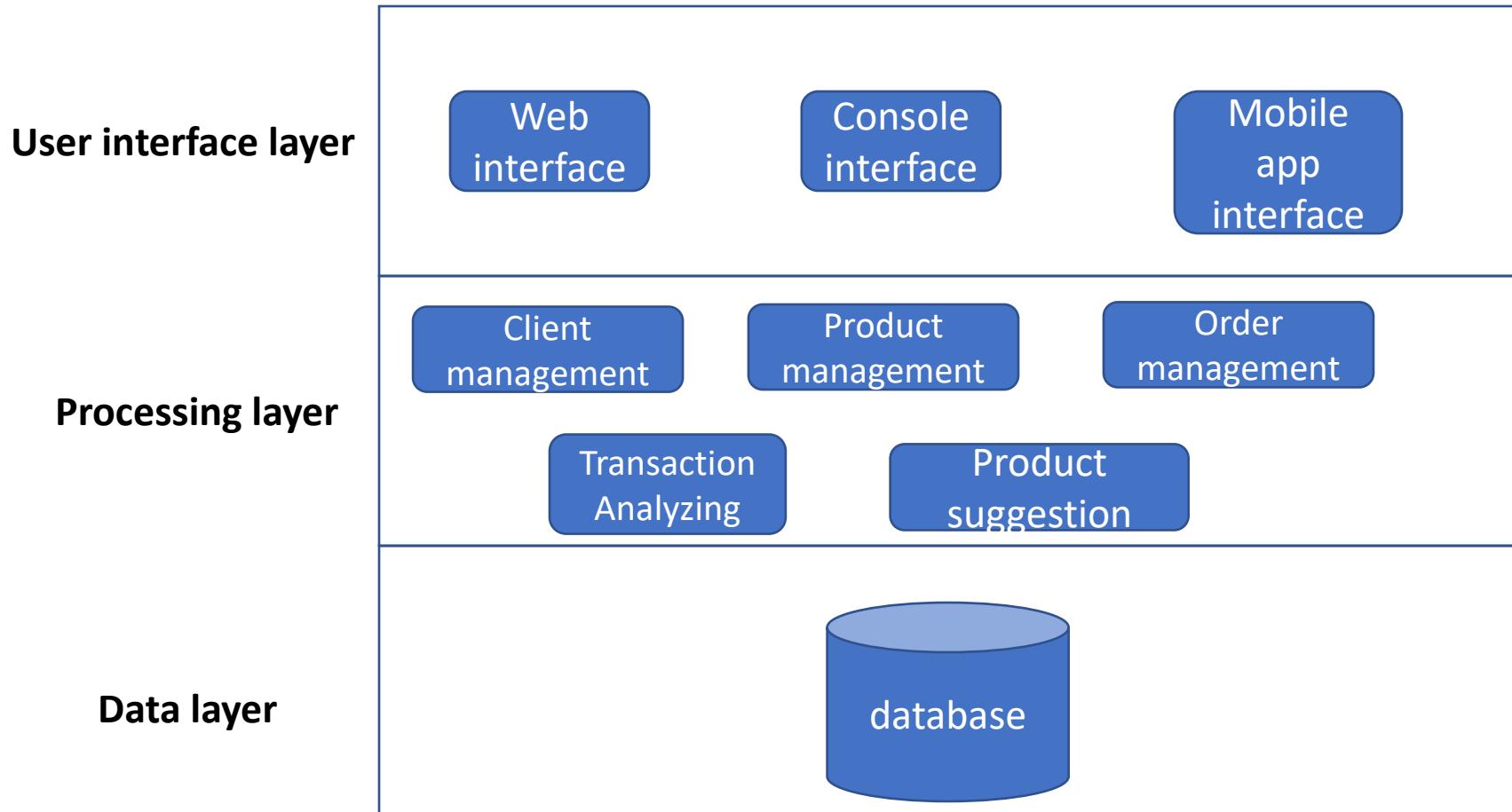


Mô hình OSI

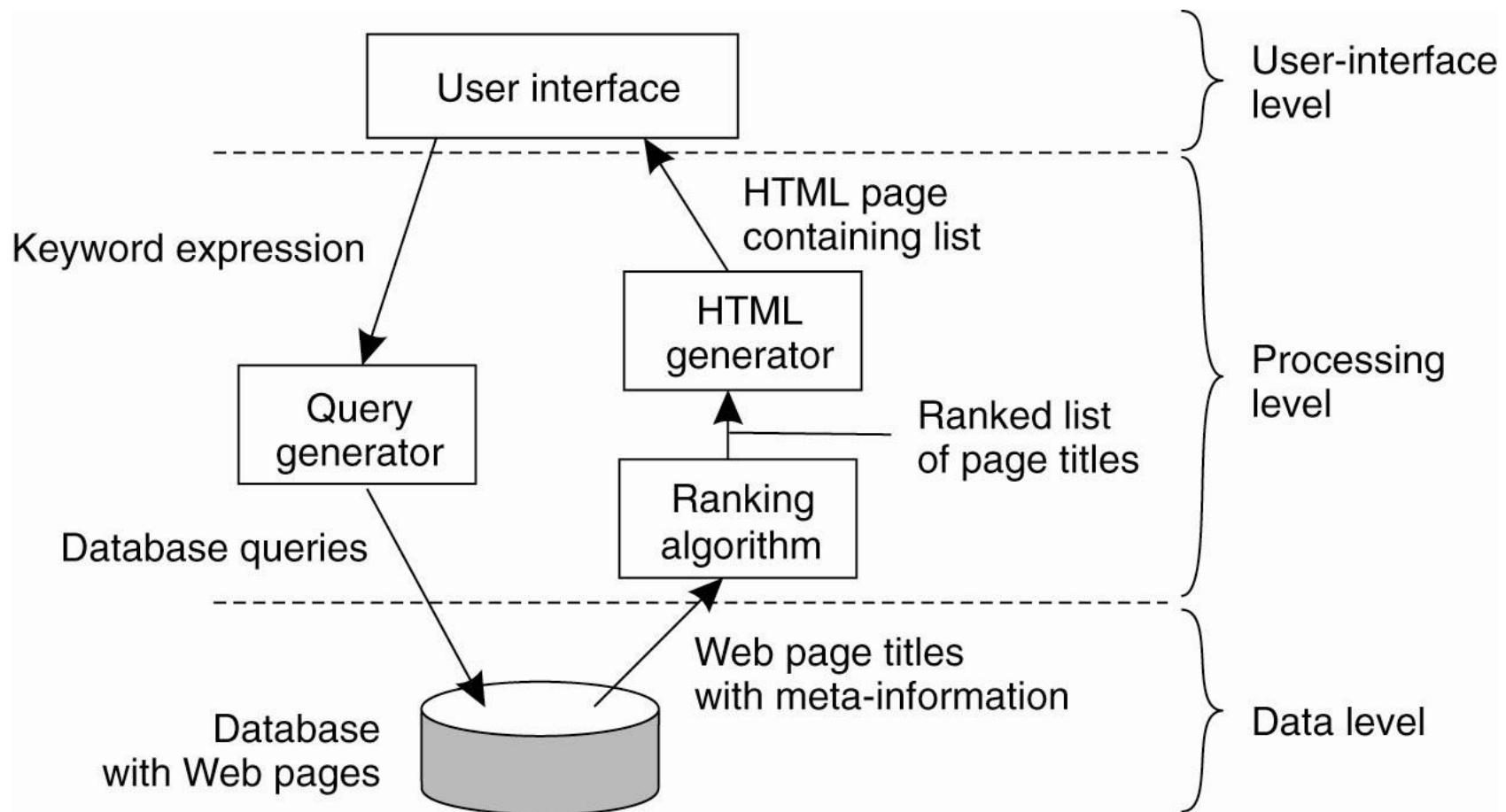


Mô hình truyền
thông thông dụng

Ví dụ 1: E-commerce system

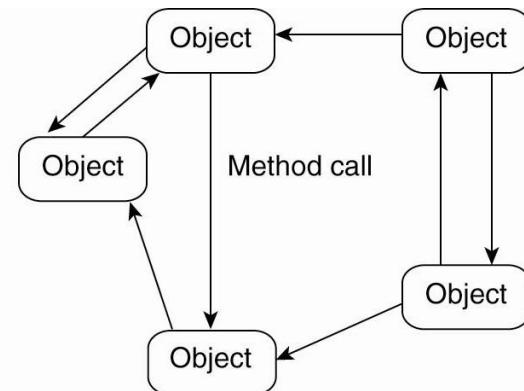


Ví dụ 2: a search engine

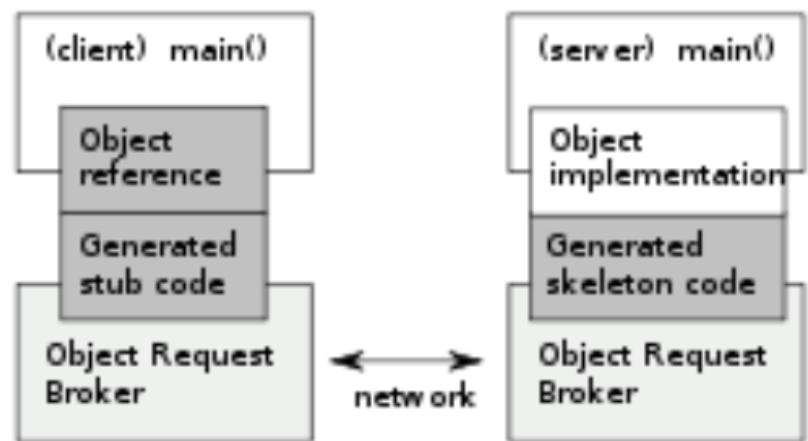


3.2.2. Kiến trúc hướng đối tượng

- Thành phần <> đối tượng
- Connector <> Lời gọi phương thức
- Object Client và Object server
- Kết nối lỏng giữa các đối tượng
- Ví dụ: CORBA (Common Object Request Broker Architecture)



(b)



Ưu nhược điểm

- **Ưu**

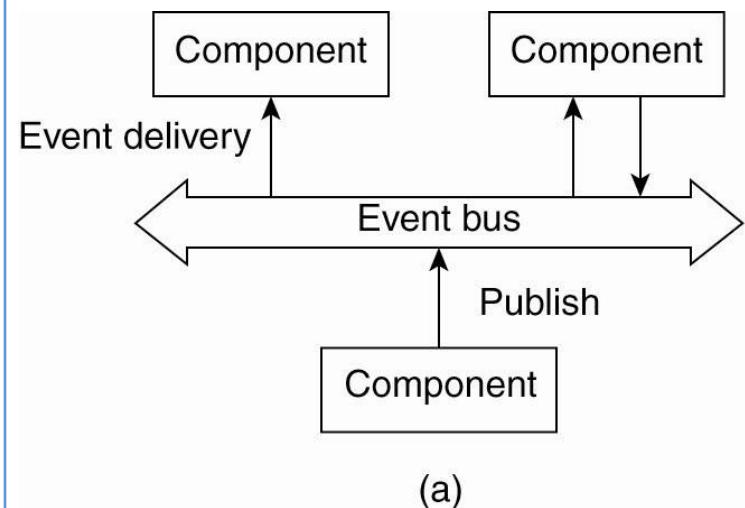
- Ánh xạ vào các đối tượng trong thế giới thật → dễ hiểu
- Dễ dàng bảo trì và nâng cấp
- Tính tái sử dụng (Polymorphism & Abstraction)
- Kiểm soát lỗi
- Mở rộng chức năng mà không ảnh hưởng hệ thống
- Dễ dàng kiểm thử với encapsulation
- Giảm thời gian và chi phí phát triển

- **Nhược**

- Khó khăn trong việc xác định các lớp, các đối tượng
- Kích cỡ chương trình lớn
- Chương trình chạy chậm hơn (so với procedure programs)
- Không phải phù hợp với mọi bài toán

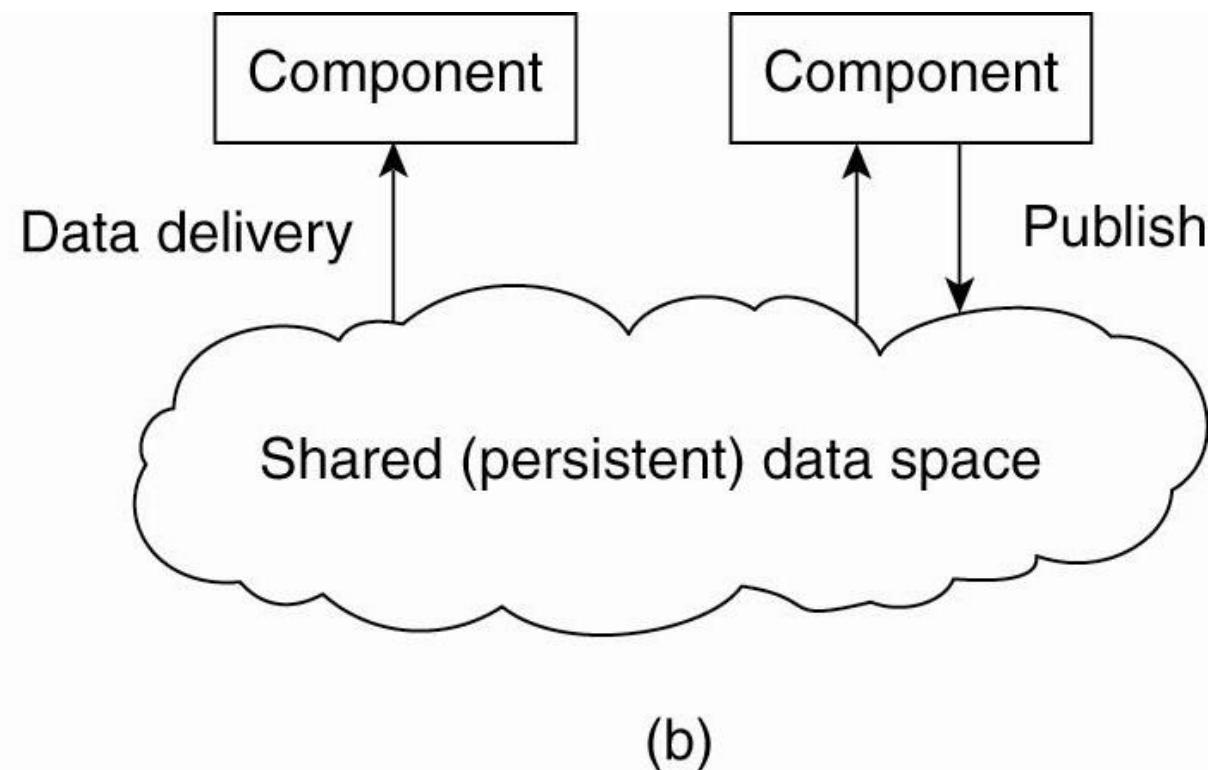
3.2.3. Kiến trúc hướng sự kiện

- Thành phần hệ thống trao đổi thông tin với nhau thông qua **các sự kiện**
- Các sự kiện chứa các thông tin cần trao đổi
- Các sự kiện có thể kích hoạt các thao tác trong các tiến trình
- Có thể thực hiện theo mô hình điểm đến hoặc mô hình trực quang bá sự kiện
- Ví dụ
 - **mô hình thuê bao/xuất bản**
- **Liên kết lỏng**



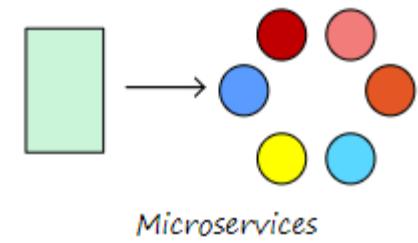
3.2.4. Kiến trúc hướng dữ liệu

- Các thành phần trao đổi thông tin thông qua kho dữ liệu chung

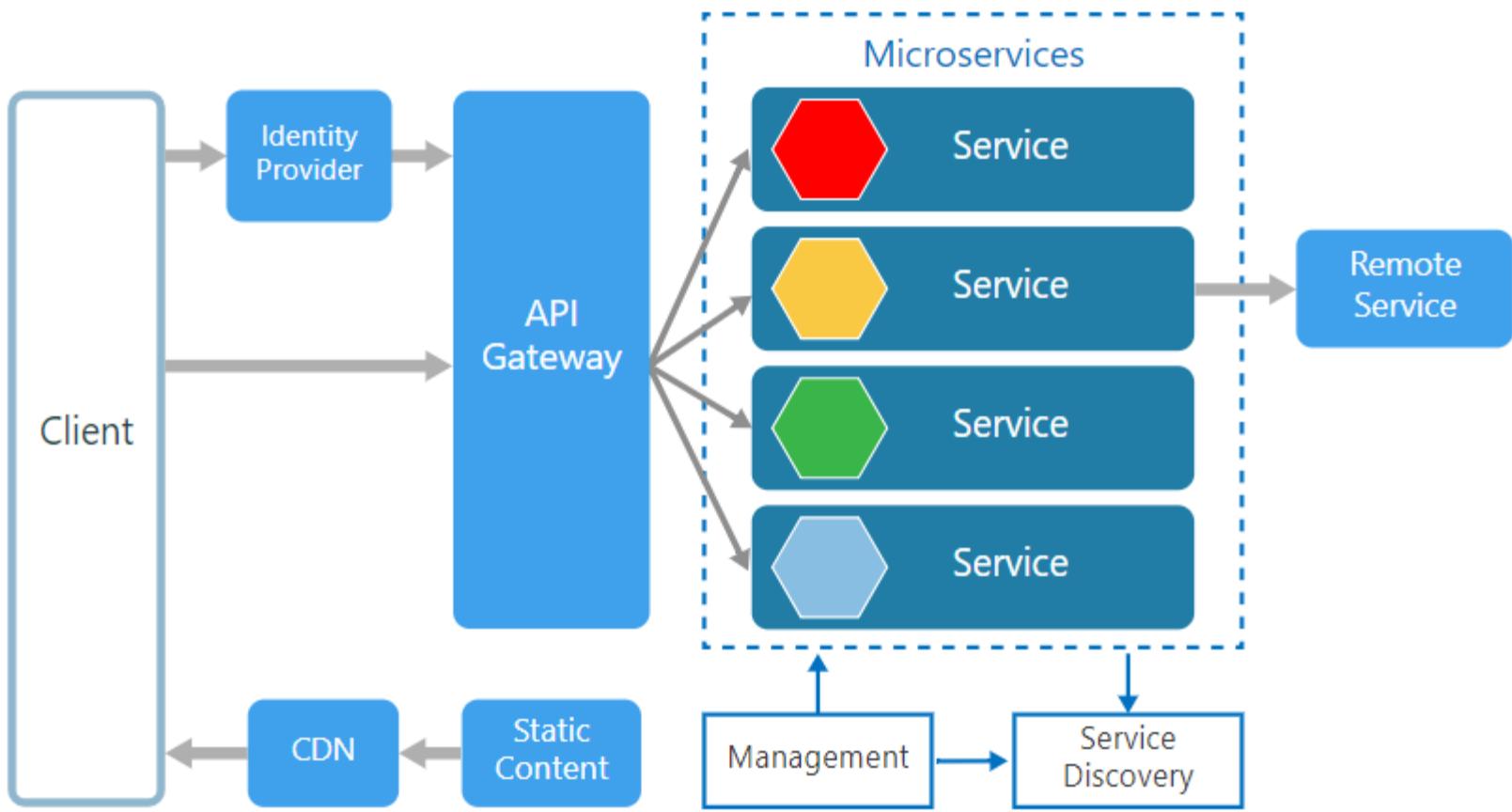


3.2.5. Microservices

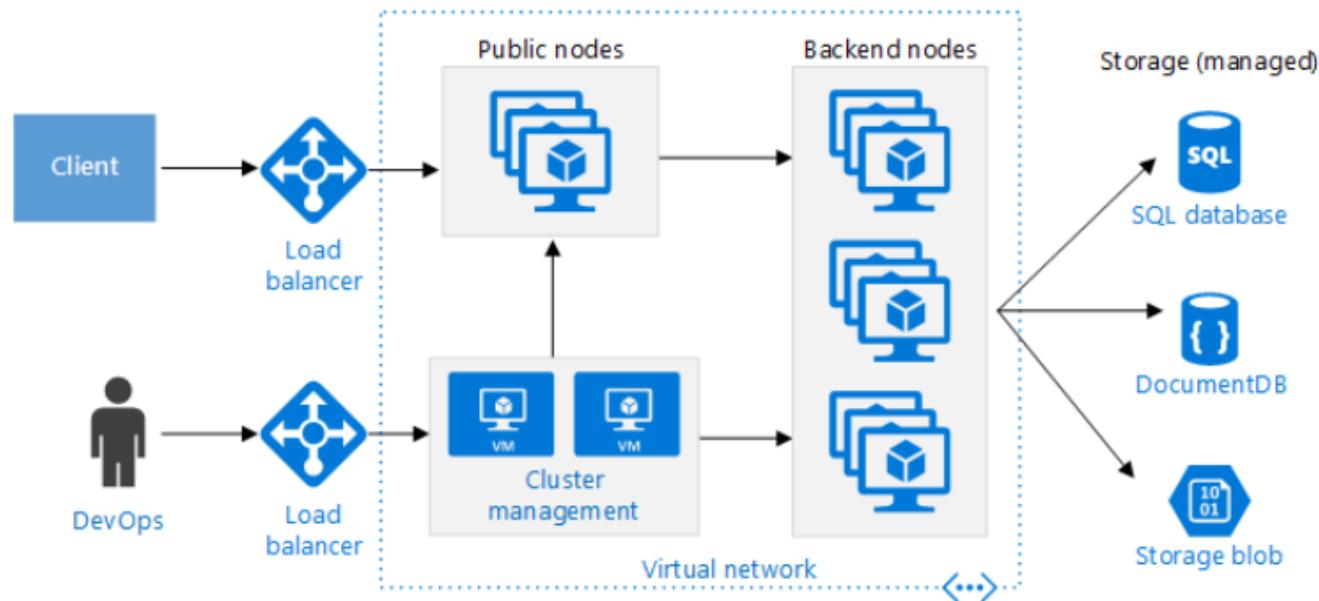
- Chuyển đổi monolithic → microservices
- Xây dựng ứng dụng dựa trên số lượng nhỏ các services, mỗi services chạy trên tiến trình riêng và hoàn toàn triển khai độc lập được.
- Ưu điểm:
 - Đơn giản triển khai
 - Đơn giản để hiểu
 - Tái sử dụng
 - Nhanh chóng cách ly thành phần hỏng
 - Giảm thiểu nguy cơ khi thực hiện thay đổi



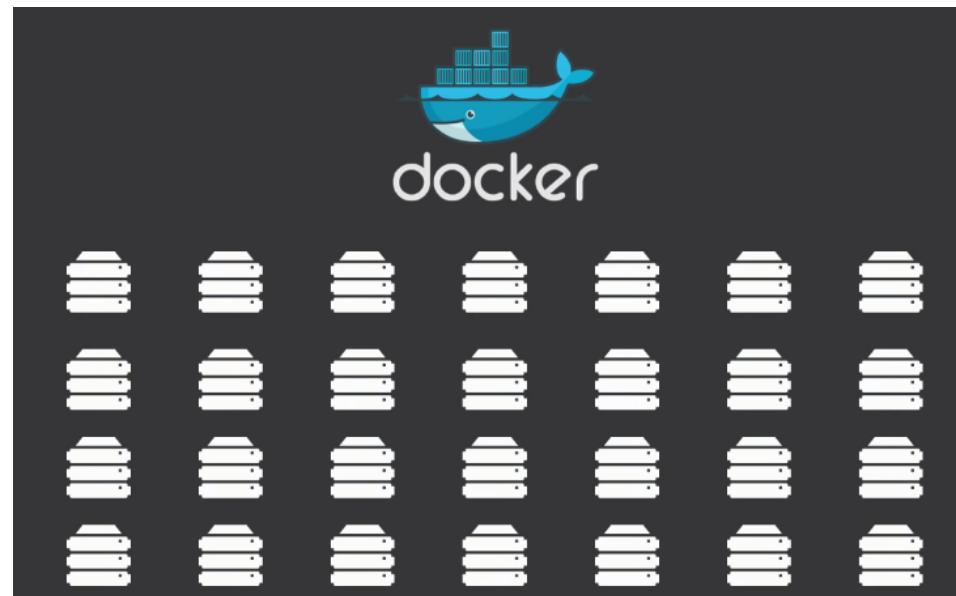
Microservices



Microservices



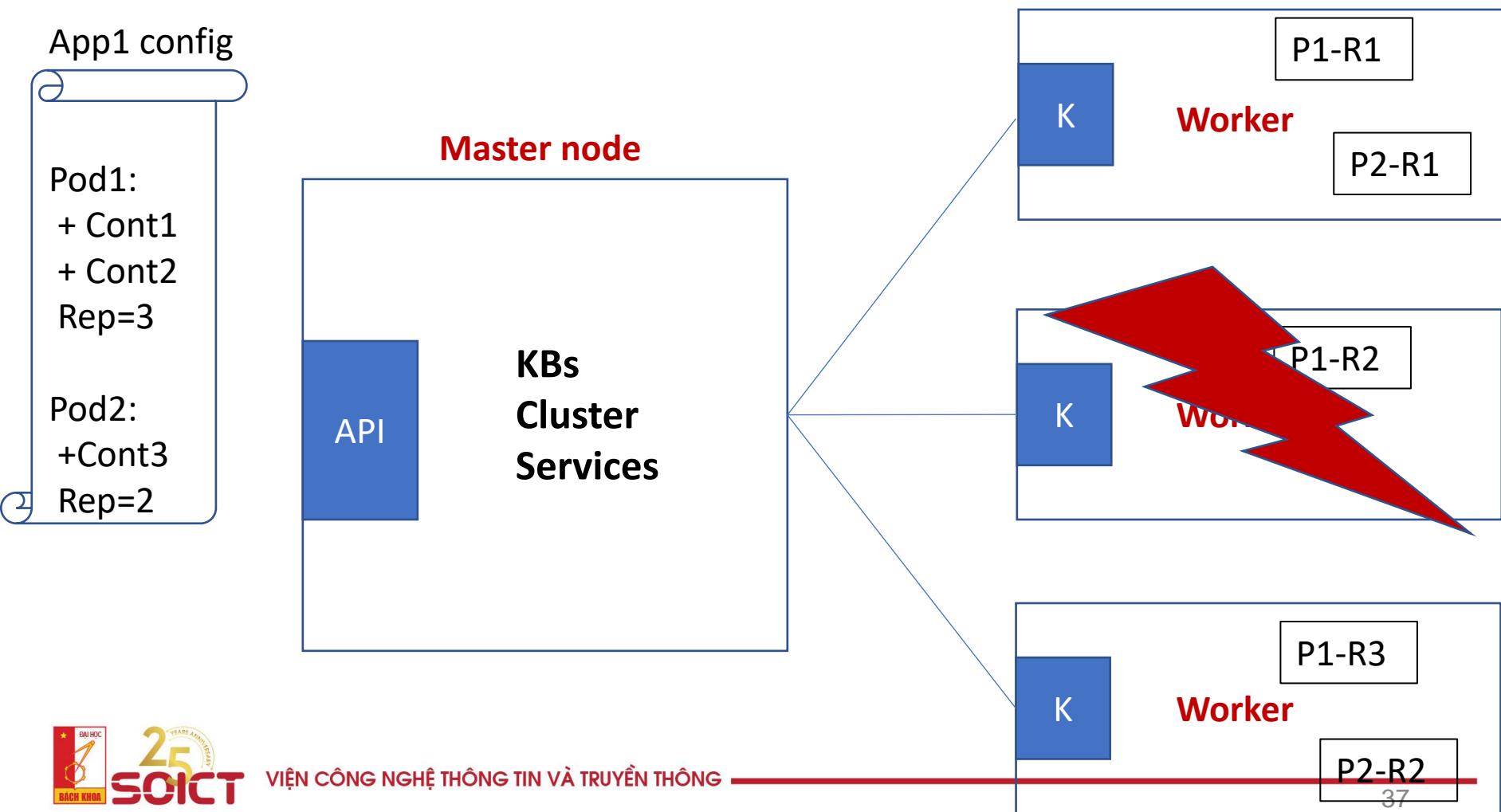
Vấn đề!!!



Container Orchestration tools

- Amazon ECS (EC2 Container Service)
- Azure Container Service (ACS)
- Cloud Foundry's Diego
- CoreOS Fleet
- Docker Swarm
- Kubernetes

Kubernetes



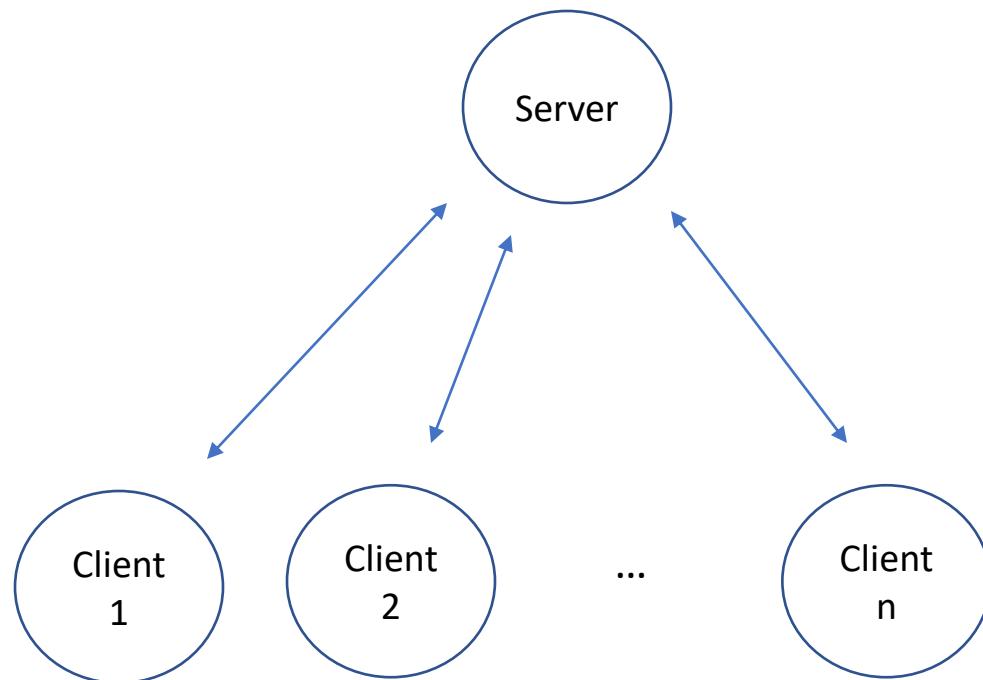
3.3. Kiến trúc hệ thống

- Kiến trúc tập trung
- Kiến trúc không tập trung
- Kiến trúc hỗn hợp

3.3.1. Kiến trúc tập trung

Kiến trúc client-server

Kiến trúc đa tầng



Kiến trúc client-server

-Client:

- gửi yêu cầu, nhận kết quả, hiển thị cho NSD

-Server:

- lắng nghe, nhận yêu cầu, xử lý, trả lời

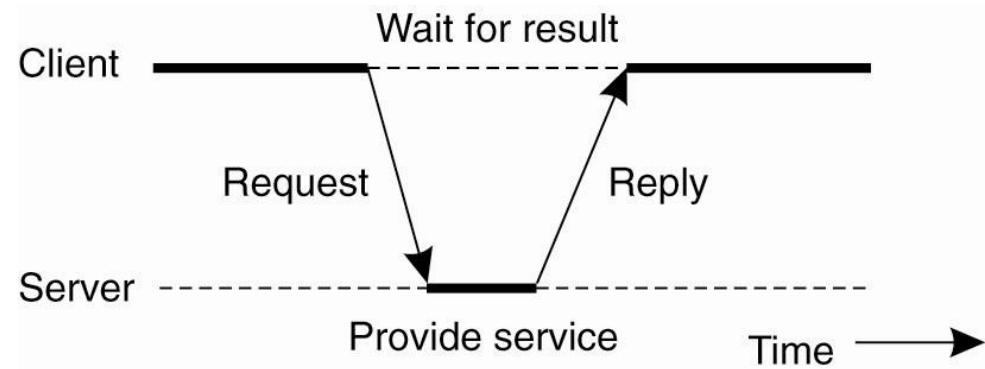
-Tương tác giữa client và server có thể là hướng kết nối hoặc không hướng kết nối

-Vấn đề

- Đăng ký server (DNS hoặc dịch vụ thư mục)

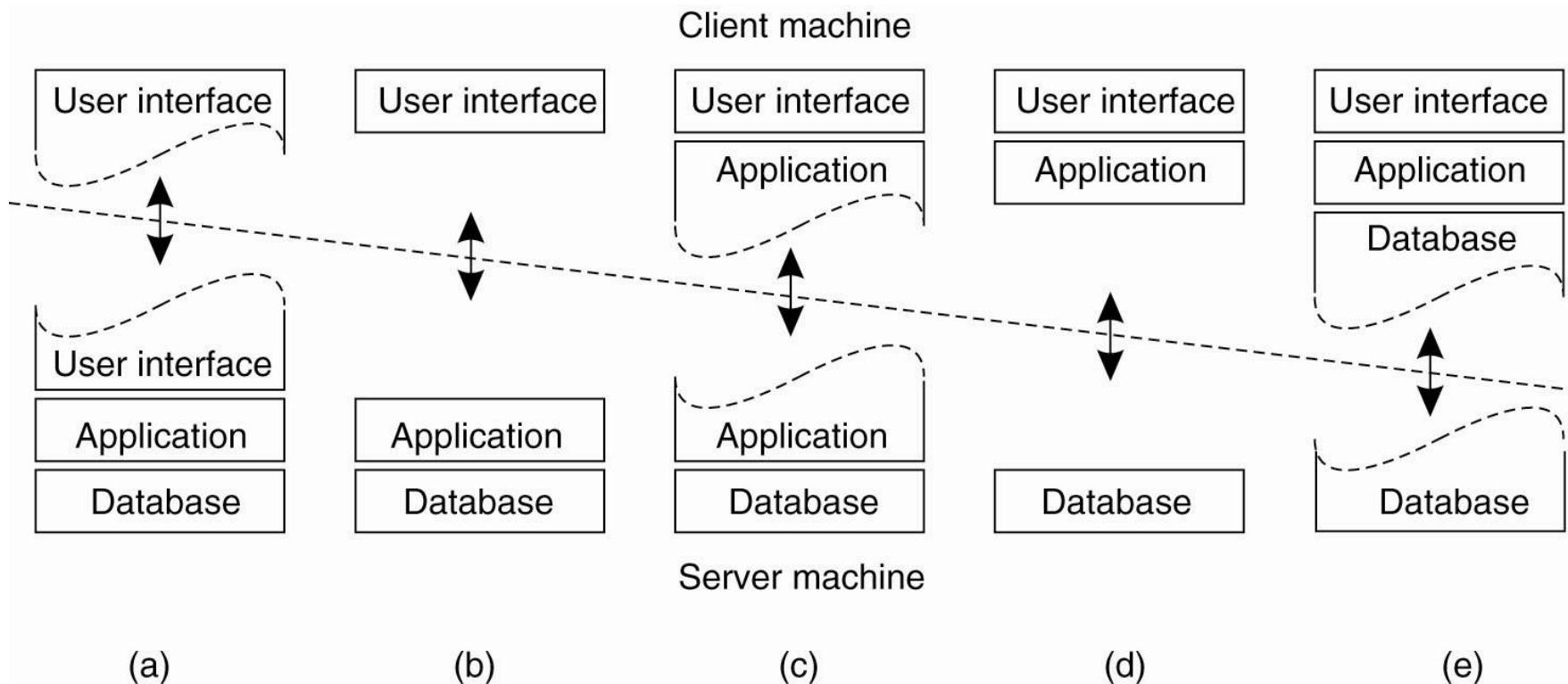
- Có thể lặp lại yêu cầu? (idempotent)

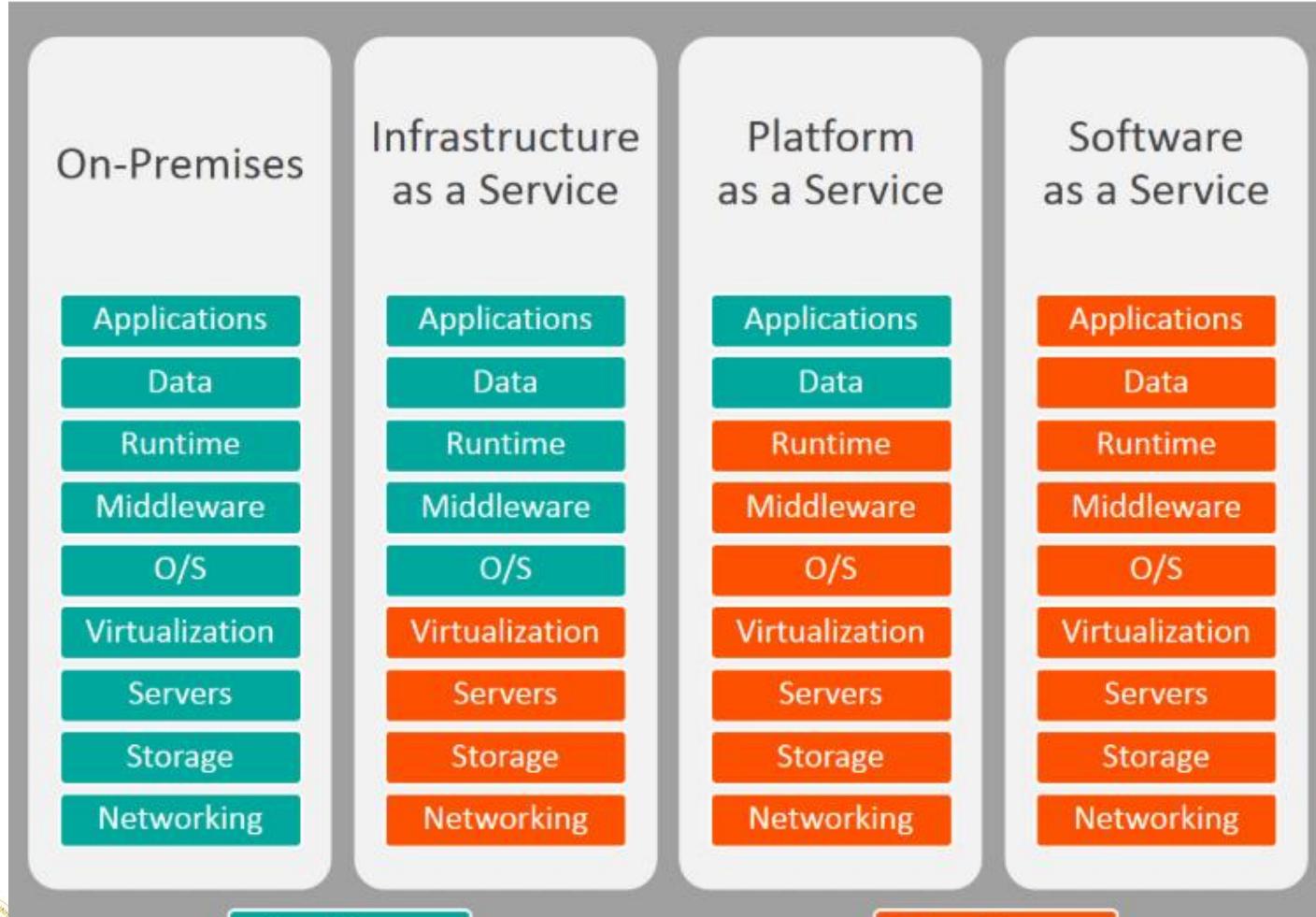
- Có bộ nhớ trạng thái?



Kiến trúc đa tầng

Các mô hình 2 bên

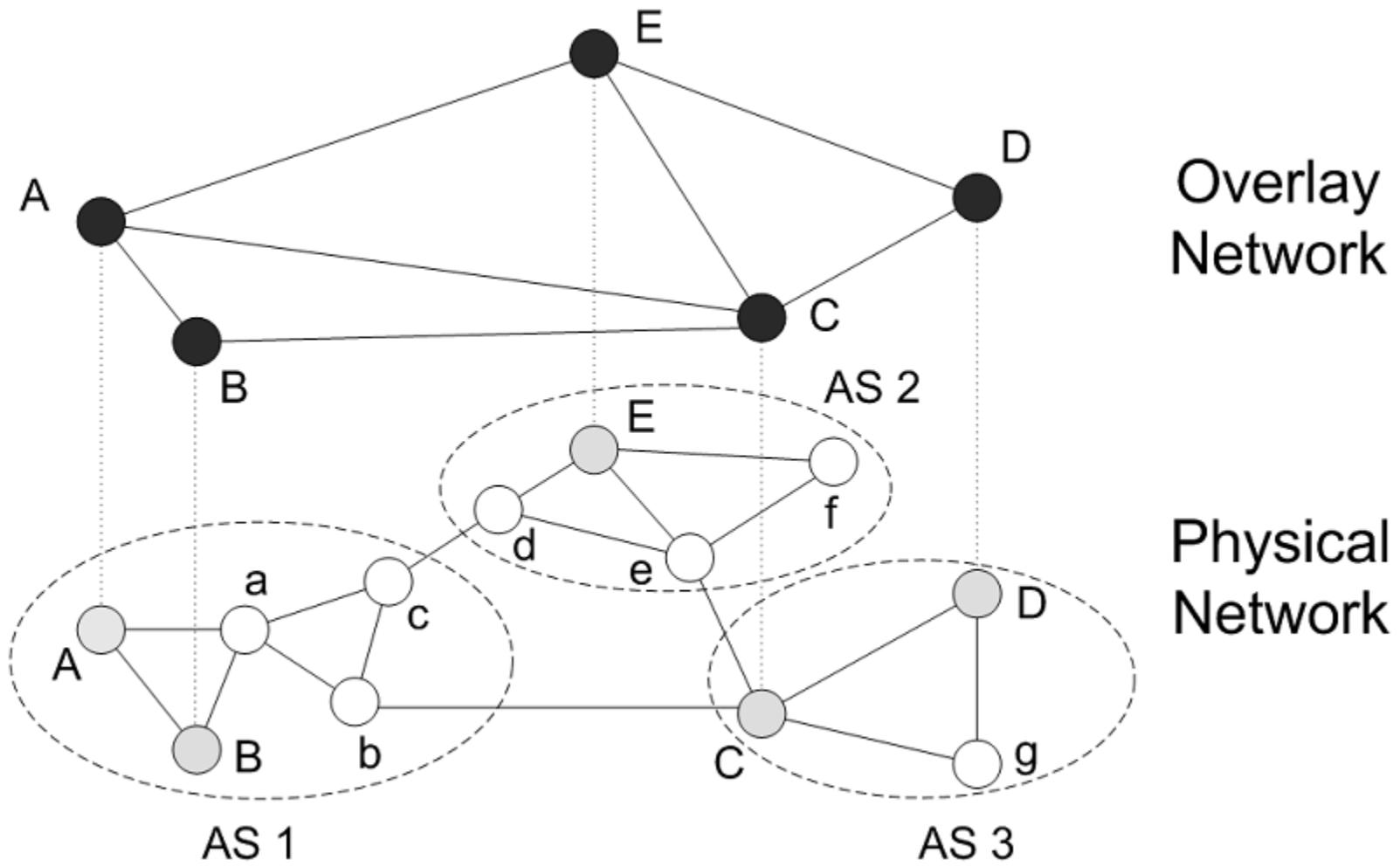




3.3.2. Kiến trúc không tập trung

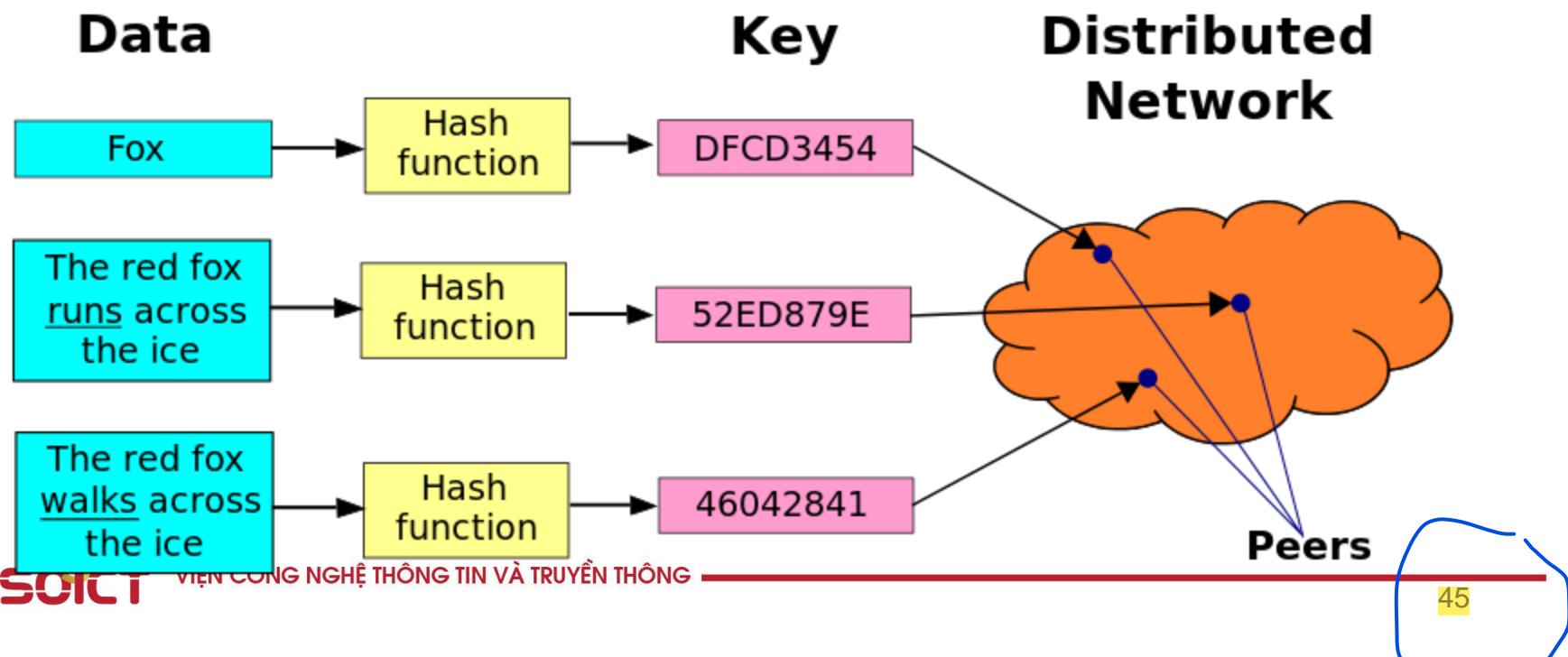
- Client và server không phân biệt vai trò
- Kết nối với nhau bằng một mạng trên mạng hạ tầng (Overlay network)
- Có **cấu trúc/Không có cấu trúc**
- P2P thuần/P2P hỗn hợp

Overlay network

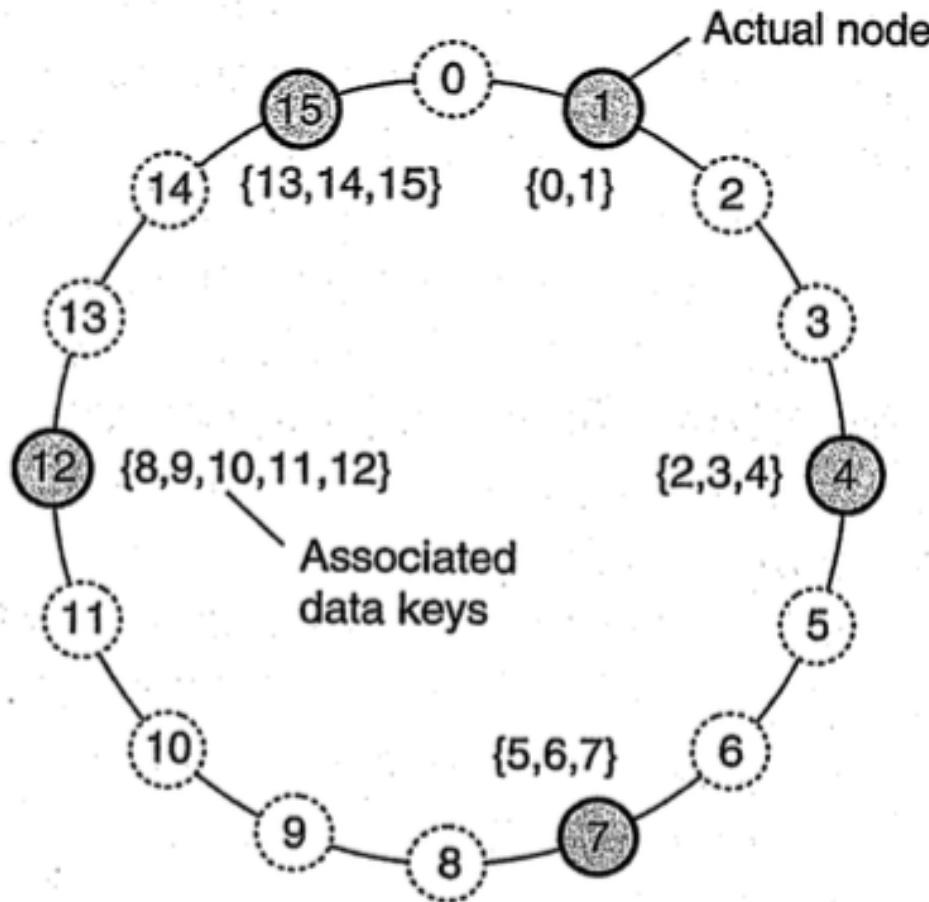


3.3.2.1. Kiến trúc P2P có cấu trúc

- Mạng overlay được xây dựng dựa trên 1 thủ tục định hình trước
- DHT (Distributed Hash Table)

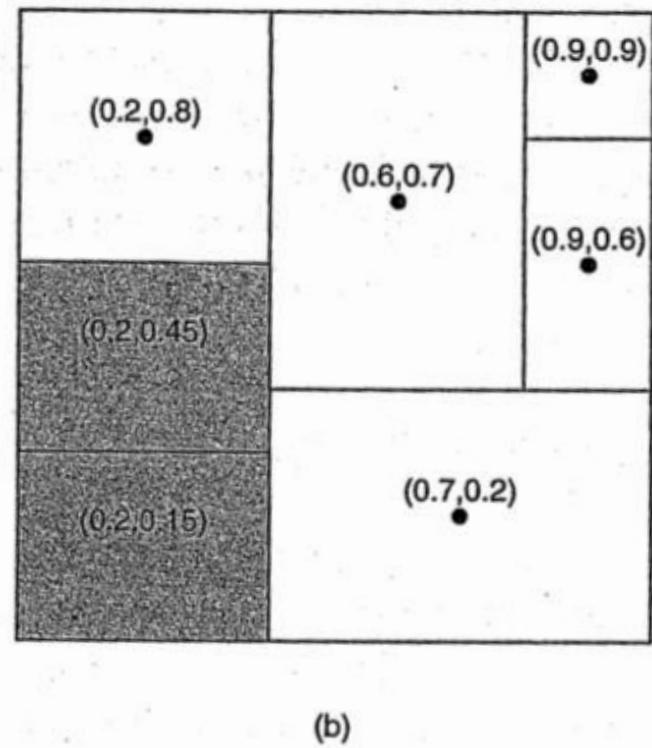
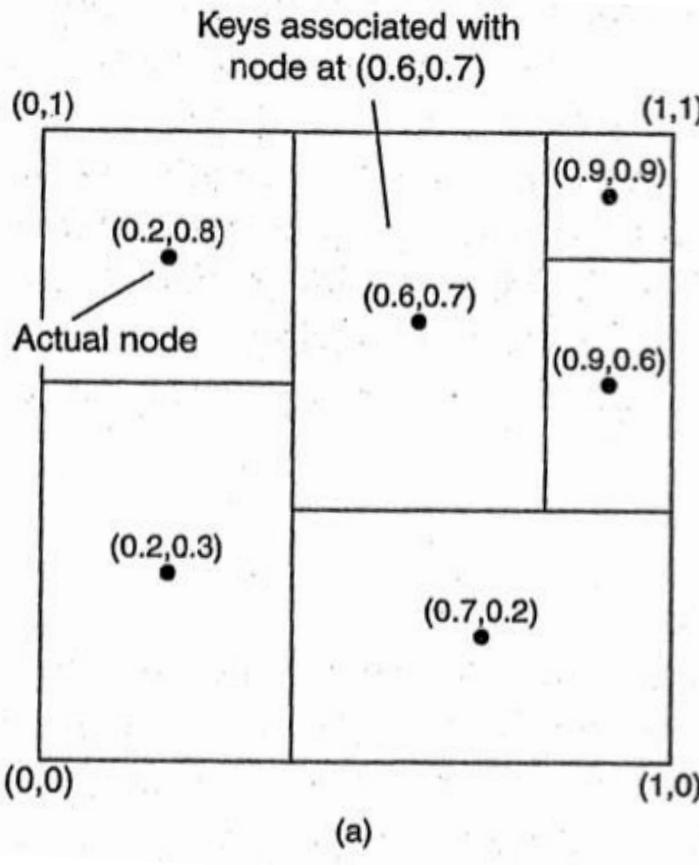


Hệ thống Chord



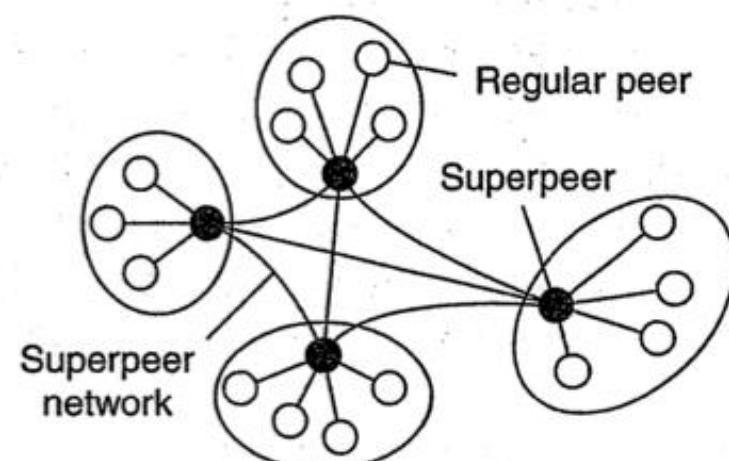
- Mạng dạng vòng
- Succ(k)
- Hàm LOOKUP(k)
- Một node muốn join hệ thống
- Một node muốn rời hệ thống

Hệ thống CAN (Content Addressable Network)



3.3.2.2. Kiến trúc P2P không có cấu trúc

- Thuật toán ngẫu nhiên để xây dựng mạng overlay (random graph).
- Mỗi node duy trì một danh sách hàng xóm (partial view).
- Dữ liệu được đưa vào hệ thống 1 cách ngẫu nhiên
- => Mỗi lần cần lấy dữ liệu ra, cần thực hiện duyệt toàn bộ hệ thống (flooding)
- =>superpeers



3.3.3. Kiến trúc hỗn hợp

- Hệ thống máy chủ biên (edge-server system)
- Hệ phân tán hợp tác

Hệ thống máy chủ biên

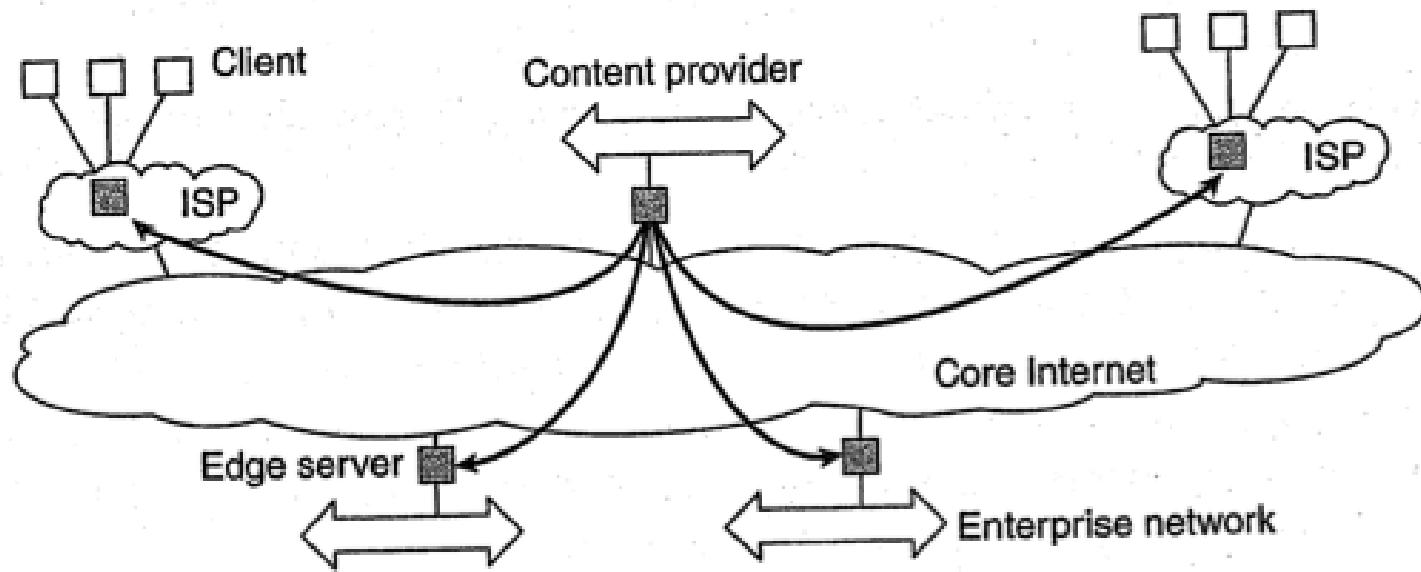
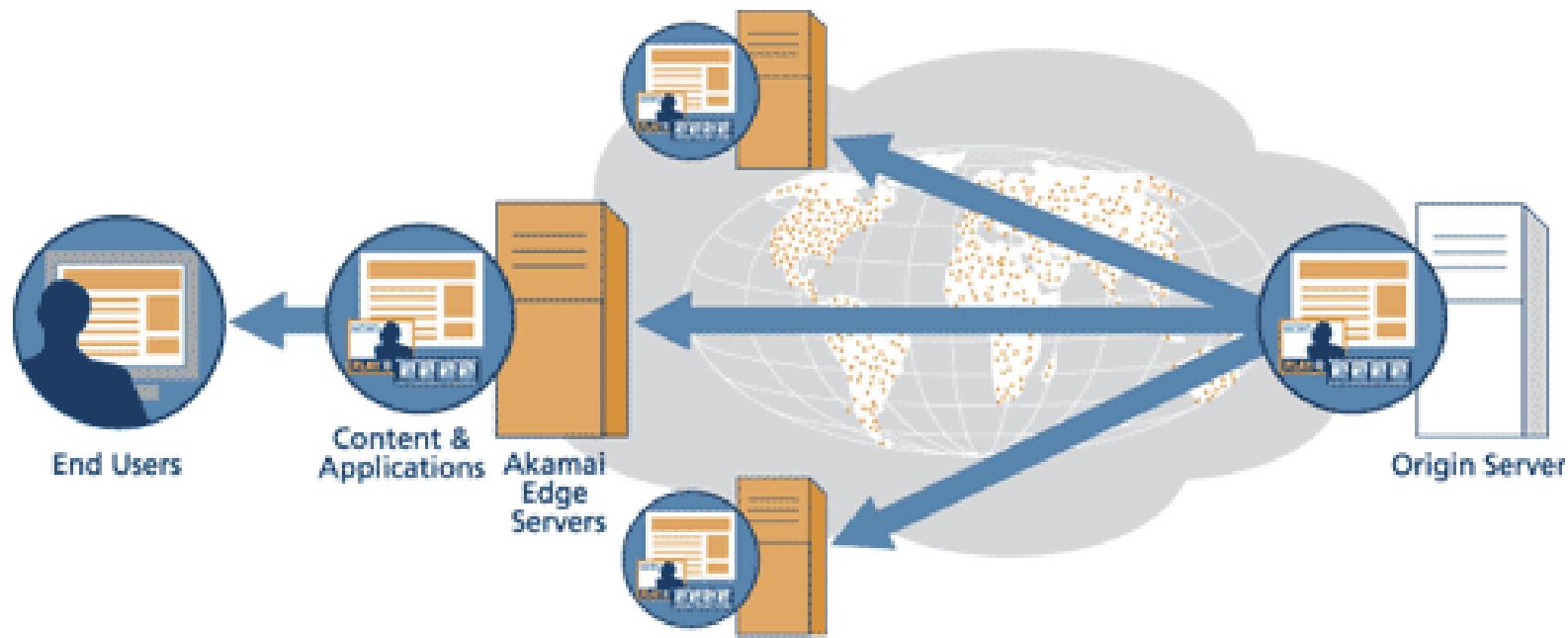
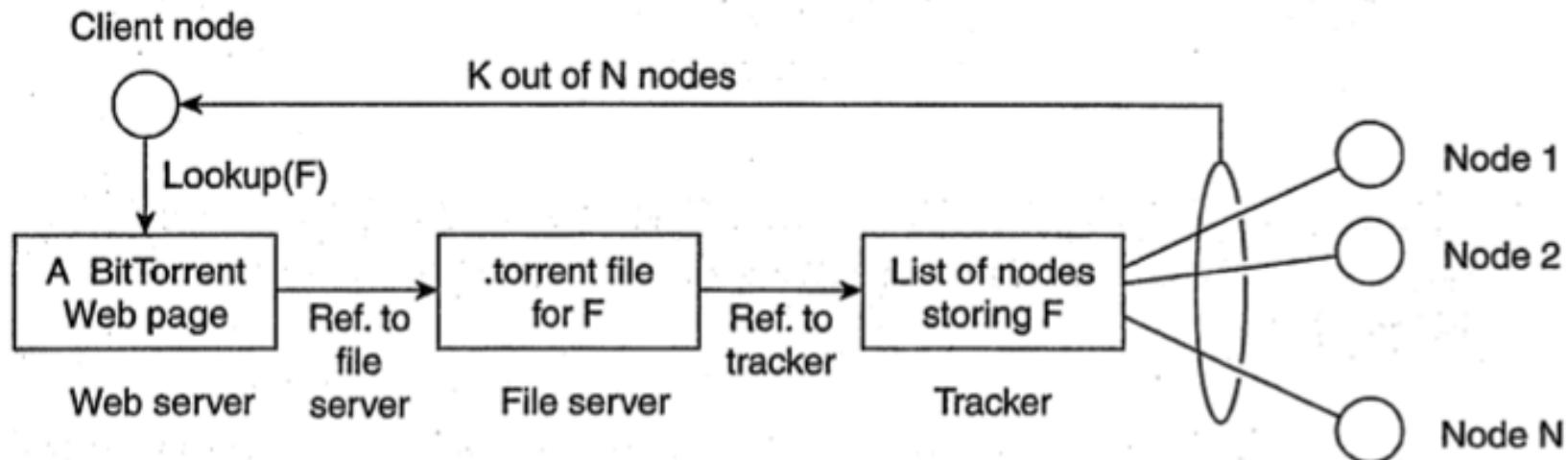


Figure 2-13. Viewing the Internet as consisting of a collection of edge servers.

VD: Content Delivery Network

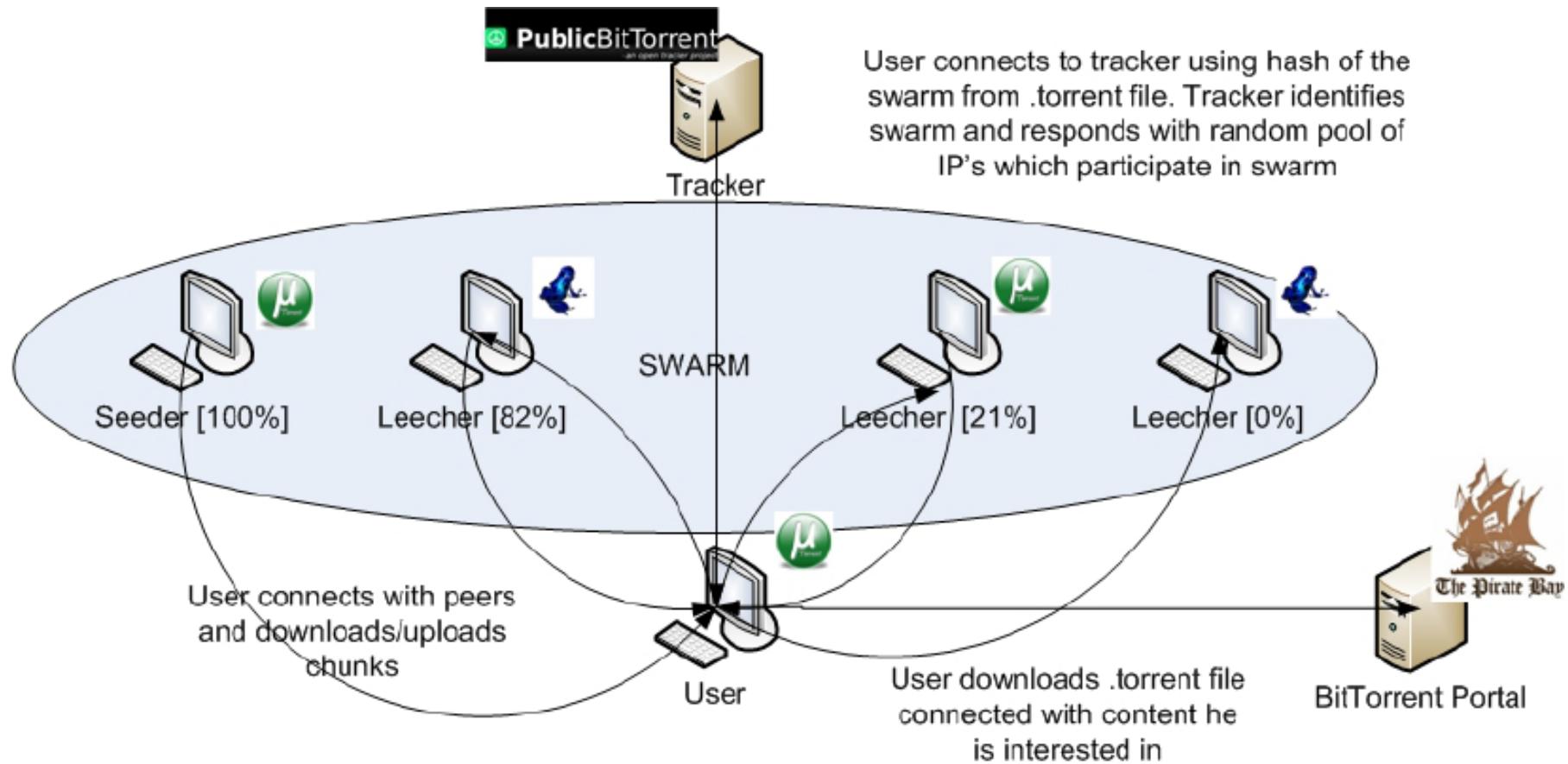


Hệ phân tán hợp tác



Hệ thống chia sẻ file BitTorrent

VD: Hệ thống BitTorrent





ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

4. Middleware trong Hệ Phân Tán

4.1. DOS, NOS

4.2. Phần mềm trung gian (Middleware)

4.1. Phần mềm hệ phân tán

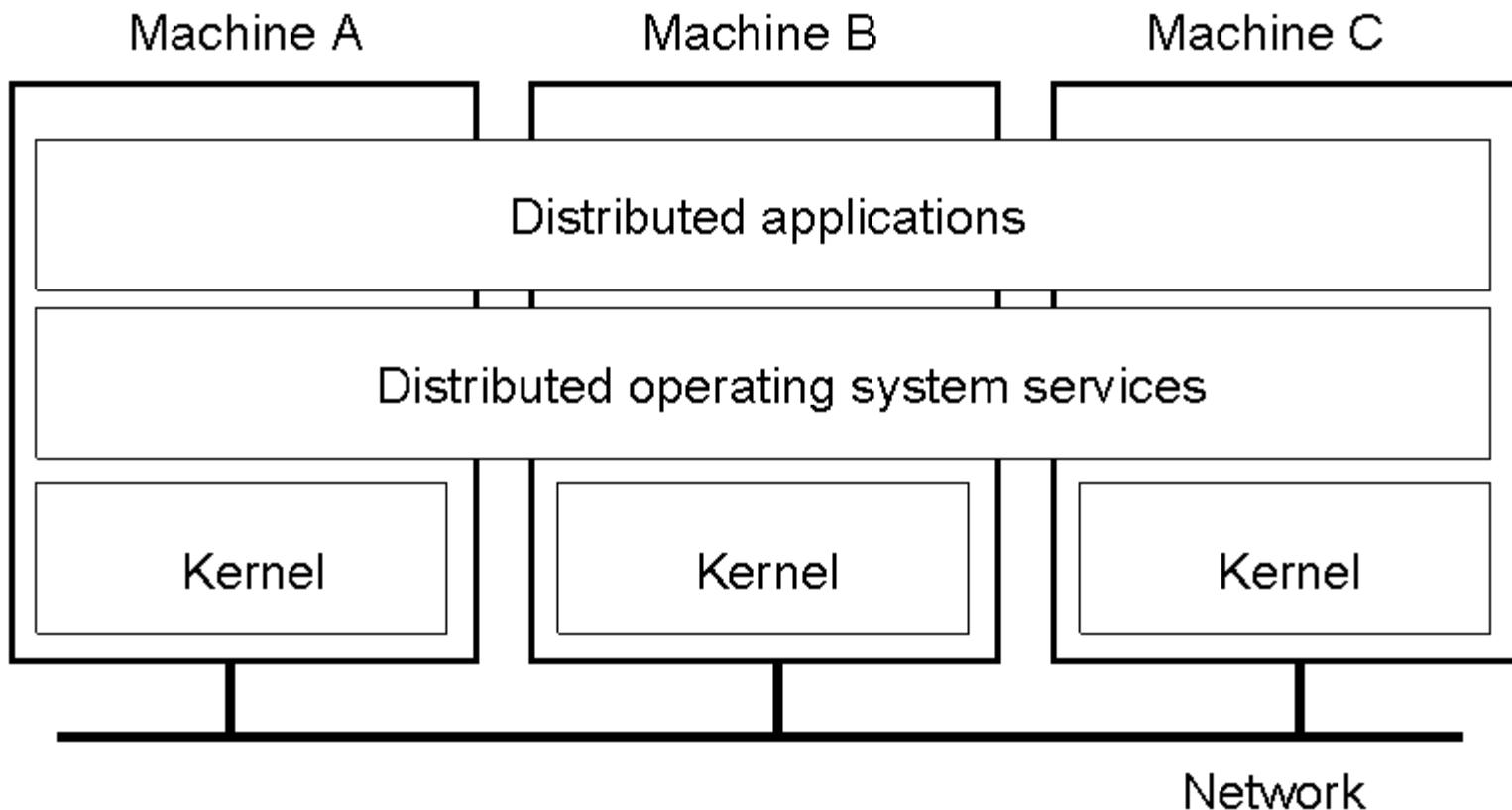
System	Description	Main Goal
DOS	OS gắn chặt với hệ thống phần cứng (máy đa vi xử lý hoặc máy tính đồng bộ) multicomputers	Trong suốt
NOS	NOS trên các máy tính cục bộ	Cung cấp dịch vụ cục bộ cho các máy tính khác
Middleware	Cài đặt các dịch vụ cơ bản để thực hiện, phát triển các ứng dụng	Tính trong suốt phân tán

- Điểm chung giữa HPT và HĐH
 - Quản lý tài nguyên
 - Che giấu tính phức tạp và tính không đồng nhất
- Có 2 loại:
 - tightly-coupled systems (DOS)
 - loosely-coupled systems (NOS)

4.1.1. Distributed Operating Systems (DOS)

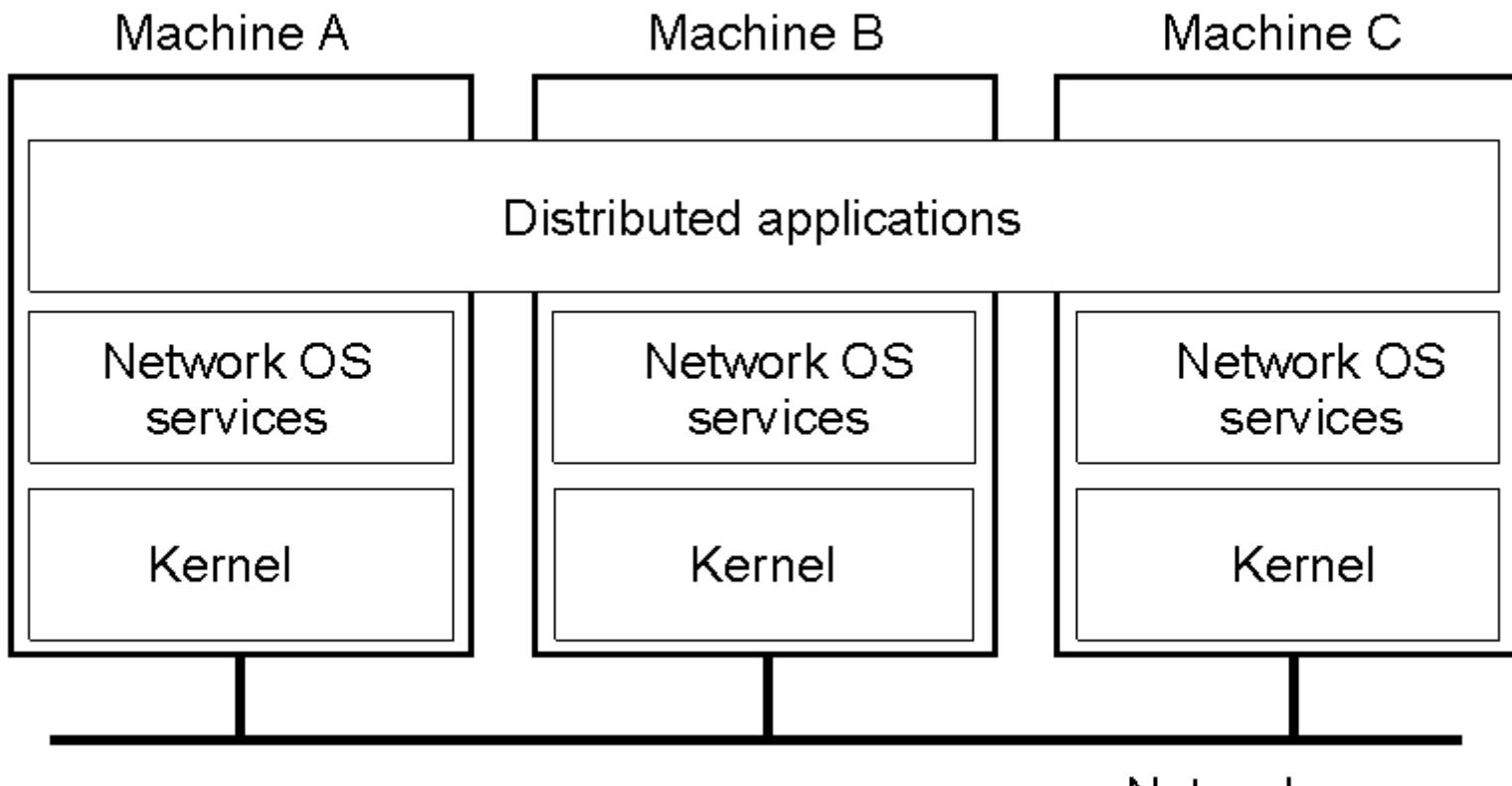
- Multiprocessor OS: quản lý tài nguyên cho đa vxl
- Multicomputer OS: HĐH dành cho hệ thống máy tính đồng nhất.
 - Vd: Inferno OS, Plan9
- Giống với HĐH đơn vxl, trừ việc xử lý nhiều CPUs

Multicomputer DOS

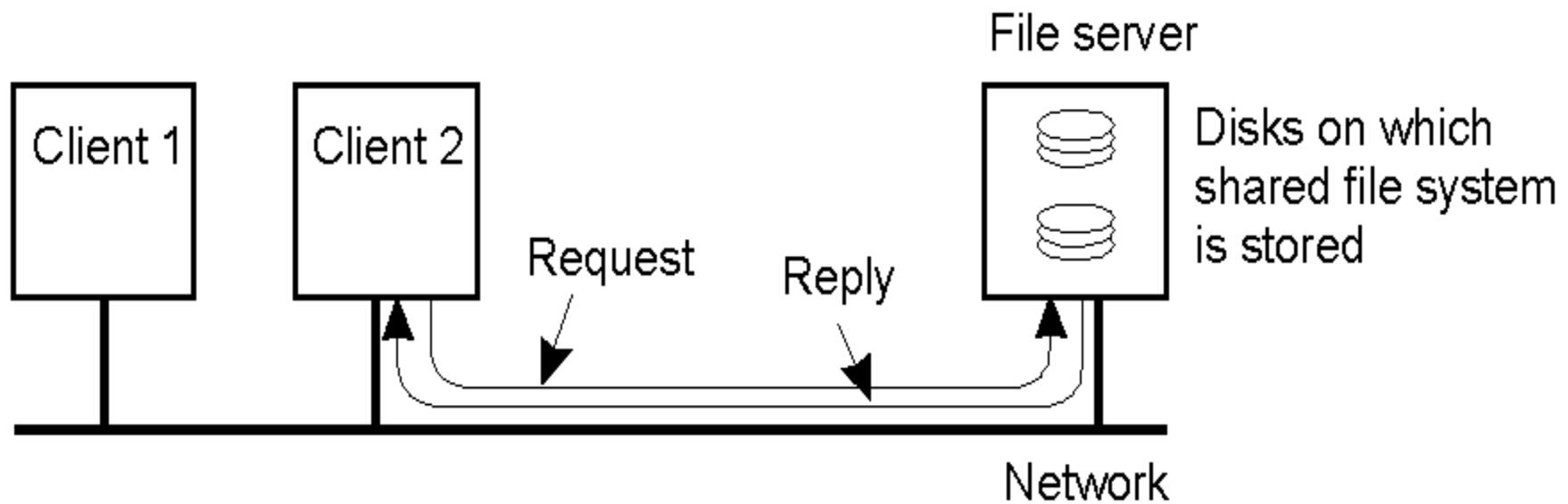


4.1.2. Hệ điều hành mạng (NOS)

- Hỗn hợp tách với hệ điều hành mạng

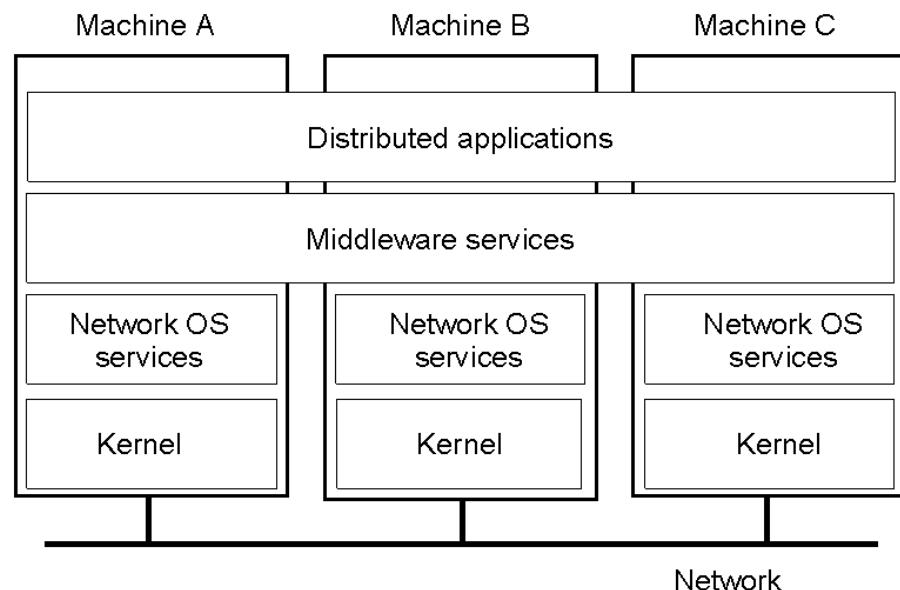


VD: Dịch vụ quản lý tệp



4.2. Middleware

- Kết hợp ưu điểm của DOS và NOS
- Middleware
- Các mô hình Middleware:
 - Mô hình quản lý file trong UNIX
 - RPC
- Dịch vụ của Middleware: truy cập trong suốt, các phương tiện trao đổi thông tin bậc cao, dvu định danh, dvu lưu trữ bền vững, v.v...



So sánh các phần mềm của hệ phân tán

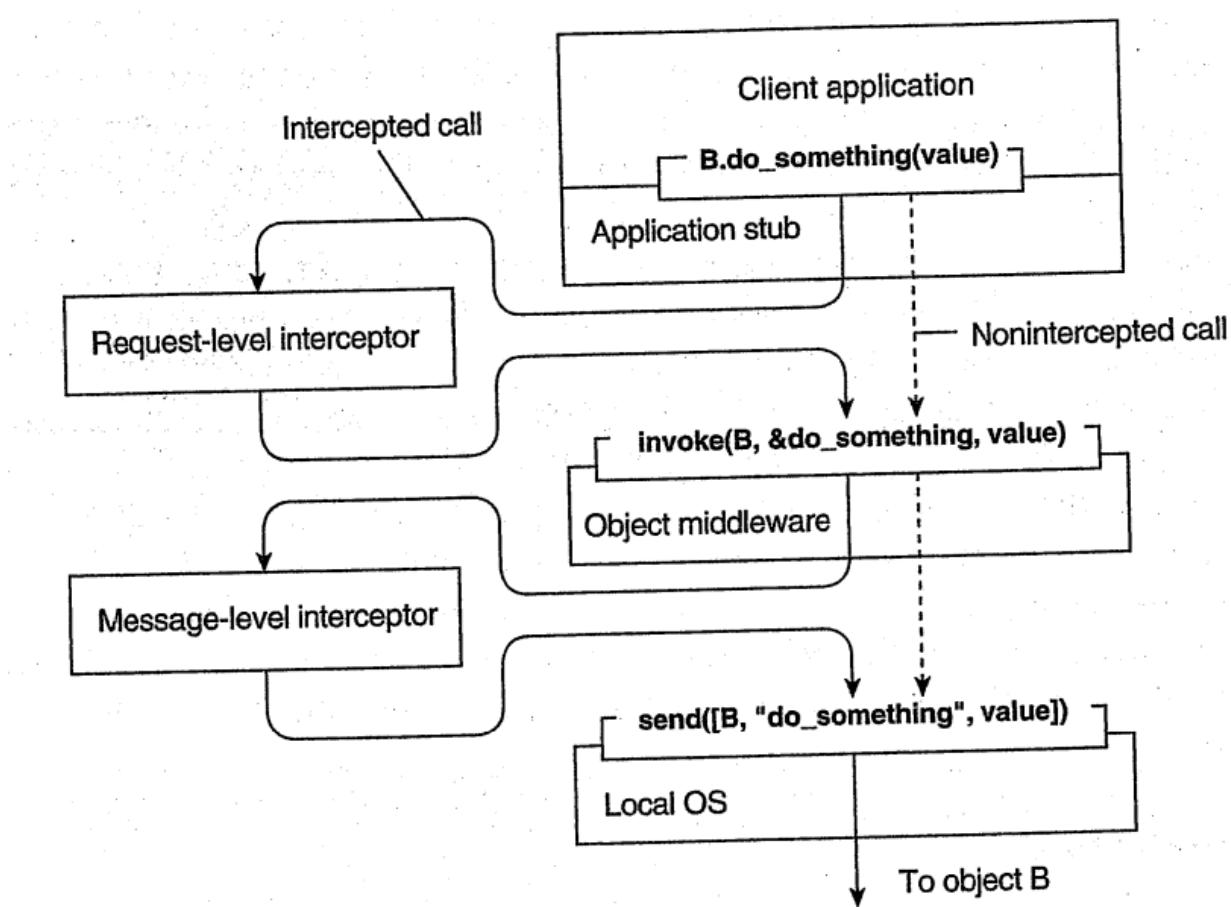
Item	Distributed OS		Network OS	Middleware-based OS
	Multiproc.	Multicomp.		
Mức độ trong suốt	Rất cao	Cao	Thấp	Cao
Một HĐH trên các nút	Yes	Yes	No	No
Số lượng bản HĐH	1	N	N	N
Trao đổi thông tin	Bộ nhớ chia sẻ	Chuyển thông báo	Tệp	Tùy thuộc
Quản lý tài nguyên	Toàn cục tập trung	Toàn cục phân tán	Theo nút	Theo nút
Co giãn	Không	Có thể	Có	Tùy thuộc
Mở	Đóng	Đóng	Mở	Mở

4.3. Middleware trong các kiến trúc

- Vị trí của middleware
- VD: CORBA, TIB/Rendezvous
- Ưu điểm: dễ dàng hơn cho thiết kế ứng dụng.
- Nhược điểm: không tối ưu cho mỗi nhà phát triển ứng dụng.
- Giải pháp:
 - Sử dụng nhiều phiên bản khác nhau của middleware.
 - Tách biệt cơ chế và chính sách → dễ dàng cấu hình, thích nghi và tùy chỉnh.

VD giải pháp: Interceptors

- Cấu trúc phần mềm, cho phép chặn các dòng điều khiển thông thường, cho phép các đoạn mã khác được thực thi.





25
YEARS ANNIVERSARY
SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Câu hỏi?

