

### Câu 1:

#### 1. Hai đặc điểm quan trọng nhất của hệ thống phân tán

##### A. Tính minh bạch (Transparency)

- Là khả năng hệ thống che giấu bản chất phân tán đối với người dùng và lập trình viên.
- Giúp người dùng thao tác với hệ thống như thể nó là một hệ thống đơn.
- Bao gồm nhiều dạng như: minh bạch truy cập, minh bạch vị trí, minh bạch lỗi, minh bạch di trú, v.v

##### B. Tính mở (Openness)

- Hệ thống được xây dựng dựa trên các chuẩn và giao thức mở.
- Cho phép các thành phần từ nhiều nhà cung cấp khác nhau tương tác, nâng cấp, mở rộng linh hoạt.
- Hỗ trợ khả năng tương tác (interoperability) và tái sử dụng dịch vụ hoặc tài nguyên.

#### 2. Ba lý do khiến ứng dụng phân tán phức tạp hơn ứng dụng đơn lẻ

##### A. Truyền thông qua mạng không ổn định

- Dữ liệu và thông điệp giữa các node truyền qua mạng (TCP/IP), nên dễ gặp lỗi:
  - Mất kết nối, độ trễ cao, gói tin bị mất.
- Gây khó khăn cho việc đồng bộ dữ liệu và phản hồi người dùng đúng thời gian.

##### B. Quản lý lỗi và phục hồi phức tạp

- Một node có thể bị lỗi nhưng các node khác vẫn hoạt động.
- Phải thiết kế hệ thống chịu lỗi (fault-tolerant) và có cơ chế phục hồi (retry, replication).
- Việc phát hiện lỗi không dễ do không phân biệt được giữa lỗi và chậm trễ mạng.

##### C. Đảm bảo tính nhất quán dữ liệu

- Dữ liệu có thể được sao chép tại nhiều node khác nhau.
- Việc cập nhật đồng thời dễ gây xung đột, dữ liệu không đồng bộ.
- Giải pháp như CAP Theorem chỉ ra rằng: Không thể cùng lúc đảm bảo tính nhất quán (C), tính sẵn sàng (A), và khả năng chịu lỗi phân mảnh (P).

### Câu 2 :

#### Yếu tố ảnh hưởng đến hiệu năng hệ thống phân tán

##### A. Phần cứng

- CPU: Nếu năng lực xử lý giữa các node không đồng đều, dễ gây mất cân bằng tải.
- RAM: RAM giới hạn ảnh hưởng đến tốc độ xử lý và lưu trữ tạm thời.

- Kênh truyền: Thiết bị I/O chậm gây tắc nghẽn dữ liệu, giảm hiệu năng.

B. Mạng

- Băng thông: Băng thông thấp làm giảm tốc độ truyền dữ liệu giữa các node.
- Topology: Cấu trúc kết nối ảnh hưởng đến độ trễ, tính chịu lỗi và khả năng mở rộng.

So sánh yêu cầu quản lý tài nguyên giữa Distributed OS và Network OS

| Yếu tố             | Distributed OS                  | Network OS                            |
|--------------------|---------------------------------|---------------------------------------|
| Tầm nhìn           | Như một hệ điều hành thống nhất | Hệ điều hành riêng cho từng máy       |
| Quản lý tài nguyên | Toàn cục, tự động phân phối     | Cục bộ, chia sẻ theo yêu cầu          |
| Đồng bộ hóa        | Cần đồng bộ tiến trình, dữ liệu | Không yêu cầu đồng bộ                 |
| Phù hợp với        | Cloud, HPC, siêu máy tính       | Mạng văn phòng, chia sẻ file đơn giản |

Câu 3:

| Loại hệ thống      | Đặc điểm chính   | Mục tiêu chính                                 | Ví dụ điển hình                   |
|--------------------|--|--|-----------------------------------|
| Điện toán phân tán | Gồm nhiều máy tính phối hợp xử lý một tác vụ lớn (chia nhỏ và xử lý song song) | Tăng hiệu suất xử lý, chia sẻ tải tính toán    | Hệ thống MPI, MapReduce           |
| Thông tin phân tán | Dữ liệu được lưu trữ, quản lý ở nhiều vị trí khác nhau trên mạng               | Truy cập dữ liệu nhanh hơn, dự phòng dữ liệu   | CDN, hệ thống DNS                 |
| Lan tỏa phân tán   | Thiết bị nhỏ gọn, nhúng, hoạt động trong môi trường thực (physical world)      | Tương tác liên tục với môi trường và con người | IoT, hệ thống cảm biến môi trường |

Phân tích kiến trúc hệ thống điện toán lưới (Grid Computing)

Hệ thống điện toán lưới gồm 3 lớp chính:

1. Lớp Ứng dụng (Application Layer)

- Là nơi người dùng tương tác với hệ thống, thực thi các tác vụ như mô phỏng, tính toán khoa học, phân tích dữ liệu.
- Cung cấp giao diện thân thiện, có thể là CLI hoặc GUI.
- Đóng vai trò yêu cầu và xử lý kết quả.

## 2. Lớp Trung gian (Middleware Layer)

- Là lớp “trung gian thông minh” giữa ứng dụng và tài nguyên phần cứng.
- Các chức năng chính:
  - Lập lịch (scheduling): phân phối công việc cho các node.
  - Bảo mật: xác thực người dùng, mã hóa dữ liệu.
  - Dịch vụ truyền thông: truyền dữ liệu qua các mạng không đồng nhất.
  - Ảo hóa tài nguyên: giúp ứng dụng sử dụng tài nguyên như một hệ thống thống nhất.

## 3. Lớp Tài nguyên (Resource Layer)

- Gồm các thiết bị phần cứng thực tế: CPU, bộ nhớ, lưu trữ, thiết bị ngoại vi.
- Quản lý cấp thấp (low-level): khởi tạo tiến trình, giám sát trạng thái node, cấp phát bộ nhớ.
- Là nền tảng vật lý cho toàn bộ hệ thống điện toán lưới hoạt động.

### Câu 4:

#### 1. Vì sao "Tính sẵn sàng" (Availability) là mục tiêu quan trọng nhất trong hệ thống phân tán?

Tính sẵn sàng là khả năng của hệ thống tiếp tục cung cấp dịch vụ cho người dùng ngay cả khi một phần của hệ thống bị lỗi. Trong hệ thống phân tán, đây là yếu tố sống còn vì:

- Tính chất phân tán: nhiều thành phần nằm ở các vị trí địa lý khác nhau, nên nguy cơ gián đoạn cao (mạng, phần cứng, phần mềm).
- Ứng dụng thực tế yêu cầu liên tục: các hệ thống tài chính, thương mại điện tử, y tế, hoặc IoT yêu cầu luôn sẵn sàng 24/7.
- Niềm tin người dùng và doanh thu: mất dịch vụ vài phút có thể gây mất khách hàng, thiệt hại kinh tế lớn.

#### 2. So sánh 3 hình thức "Tính trong suốt" quan trọng

| Loại trong suốt                        | Giải thích  | Ví dụ  |
|--|---|--|
| <b>Truy nhập (Access Transparency)</b> | Người dùng không cần quan tâm đến cách truy cập tài | Truy cập file trên ổ đĩa cục bộ hoặc ổ đĩa mạng dùng chung |

|                                       |   |  |
|---------------------------------------|---|--|
|                                       | nguyên (giao thức, cách kết nối).   | đều qua cùng một API open() trong Python.  |
| <b>Vị trí (Location Transparency)</b> | Người dùng không cần biết tài nguyên nằm ở đâu trong mạng.                | Mở một trang web mà không biết máy chủ đặt ở quốc gia nào.                               |
| <b>Lỗi (Failure Transparency)</b>     | Hệ thống có thể phục hồi khi có lỗi xảy ra mà người dùng không nhận thấy. | Một node trong cụm server chết, hệ thống tự chuyển sang node khác mà không ngắt dịch vụ. |

3. Mối quan hệ giữa “Tính mở” và “Khả năng tương tác” trong hệ thống phân tán

- Tính mở (Openness): Hệ thống hỗ trợ các giao thức mở, chuẩn công nghiệp, và dễ mở rộng thêm thành phần mới.
- Khả năng tương tác (Interoperability): Là khả năng các thành phần khác công nghệ, khác nhà cung cấp có thể hoạt động cùng nhau nhờ các chuẩn mở.

Mối quan hệ:

- Tính mở là điều kiện tiên quyết để đạt được khả năng tương tác.
- Nếu hệ thống đóng (sử dụng chuẩn riêng), sẽ khó tích hợp thiết bị hoặc phần mềm mới.

Câu 5:

| Tiêu chí                       | Kiến trúc phân cấp (Client-Server)        | Kiến trúc ngang hàng (Peer-to-Peer - P2P)          |
|--------------------------------|---|--|
| <b>Cấu trúc tổ chức</b>        | Có máy chủ trung tâm điều phối và lưu trữ | Các nút (peer) bình đẳng, không có trung tâm       |
| <b>Hiệu suất mở rộng</b>       | Hạn chế, dễ bị quá tải ở server           | Tốt, càng nhiều peer thì khả năng phục vụ càng cao |
| <b>Tính sẵn sàng</b>           | Dễ bị gián đoạn nếu server lỗi            | Cao, do không phụ thuộc vào một nút trung tâm      |
| <b>Bảo mật &amp; kiểm soát</b> | Dễ kiểm soát và bảo vệ dữ liệu            | Khó quản lý quyền truy cập, tiềm ẩn rủi ro bảo mật |
| <b>Ví dụ điển hình</b>         | Web server, cơ sở dữ liệu                 | BitTorrent, Skype (phiên bản cũ)                   |


| Mô hình                          | Mô tả ngắn gọn                                       | Ví dụ ứng dụng                                  |
|----------------------------------|--|---|
| <b>Phân tầng (Layered model)</b> | Các lớp logic (UI, xử lý, dữ liệu) tách biệt rõ ràng | Ứng dụng web ba lớp (frontend/backend/database) |

|  |   |  |
|--|---|--|
| <b>Đối tượng phân tán (Distributed Object Model)</b> | Giao tiếp thông qua đối tượng từ xa như RMI, CORBA                      | Java RMI, hệ thống CORBA                         |
| <b>Kênh sự kiện (Event-based model)</b>              | Các thành phần gửi/nhận thông điệp bất đồng bộ qua broker hoặc queue    | Apache Kafka, MQTT trong IoT                     |
| <b>Dữ liệu tập trung (Data-Centric model)</b>        | Dữ liệu nằm tại trung tâm, các node truy vấn và xử lý dữ liệu tập trung | Hệ thống cơ sở dữ liệu tập trung, Data Warehouse |

### 3. Vai trò và tính năng của phần mềm trung gian (Middleware) trong kiến trúc khách – chủ

Vai trò chính:

Phần mềm trung gian (middleware) là lớp kết nối giữa client và server, giúp ẩn đi sự phức tạp của mạng, đồng thời đảm bảo giao tiếp và phối hợp giữa các thành phần trong hệ thống phân tán.

 Ba tính năng chính mà middleware cung cấp:

1. Minh bạch truy cập và vị trí (Access & Location Transparency):
  - Cho phép client truy cập tài nguyên từ xa giống như tài nguyên cục bộ.
2. Quản lý truyền thông và dữ liệu:
  - Tự động hóa việc gửi/nhận thông điệp, tuần tự hóa dữ liệu (serialization), xử lý lỗi mạng.
3. Hỗ trợ khả năng mở rộng và bảo mật:
  - Middleware hiện đại hỗ trợ cân bằng tải, xác thực người dùng, mã hóa dữ liệu.

Câu 6:

. Phân loại ba loại dịch vụ trong SOA

SOA phân loại dịch vụ dựa trên vai trò trong hệ thống và mức độ trừu tượng. Ba loại dịch vụ chính:

#### A. Dịch vụ cơ bản (Basic Services / Entity Services)

- Mô tả: Các dịch vụ đại diện cho các thực thể trong hệ thống, thường tương ứng với dữ liệu CRUD (Create – Read – Update – Delete).
- Chức năng chính: Truy xuất, lưu trữ và cập nhật thông tin từ cơ sở dữ liệu.
- Ví dụ:
  - CustomerService: quản lý thông tin khách hàng (tên, địa chỉ, số điện thoại...).
  - ProductCatalogService: cung cấp danh sách sản phẩm.

---

#### B. Dịch vụ tích hợp (Composite / Integration Services)

- Mô tả: Kết hợp nhiều dịch vụ cơ bản để thực hiện một quy trình nghiệp vụ phức tạp hơn.
- Chức năng chính: Đóng vai trò là cầu nối, sử dụng orchestration (điều phối tuần tự) hoặc choreography (phối hợp phi tập trung).
- Ví dụ:
  - OrderManagementService: tích hợp CustomerService, InventoryService, PaymentService để xử lý đơn hàng.

### C. Dịch vụ quy trình (Process Services)

- Mô tả: Mô hình hóa toàn bộ quy trình nghiệp vụ, sử dụng các công cụ BPM (Business Process Management).
- Chức năng chính: Thực thi luồng nghiệp vụ đầy đủ, có thể bao gồm cả tích hợp người dùng và logic ra quyết định.
- Ví dụ:
  - LoanApprovalProcessService: kiểm tra hồ sơ, phê duyệt tín dụng, gửi thông báo.
  - EmployeeOnboardingProcessService: từ tuyển dụng đến cấp tài khoản và phân quyền.

| Giai đoạn                          | Nội dung chính  | Thách thức chính  |
|------------------------------------|---|---|
| <b>1. Phân tích &amp; thiết kế</b> | Xác định dịch vụ cần thiết, đặc tả WSDL hoặc OpenAPI, phân ranh giới dịch vụ. | Xác định đúng độ tách biệt, tránh thiết kế dịch vụ quá nhỏ hoặc quá phức tạp. |
| <b>2. Phát triển (Implement)</b>   | Lập trình dịch vụ (SOAP, REST, gRPC...), xây dựng logic và giao tiếp.         | Quản lý phụ thuộc giữa các dịch vụ, đảm bảo tái sử dụng.                      |
| <b>3. Kiểm thử (Testing)</b>       | Kiểm thử chức năng, hiệu năng, bảo mật, kiểm thử tích hợp.                    | Kiểm thử dịch vụ phụ thuộc hoặc không khả dụng, dữ liệu test thực tế.         |
| <b>4. Triển khai (Deployment)</b>  | Đưa dịch vụ lên môi trường staging/production, tích hợp với registry.         | Đồng bộ phiên bản, CI/CD, rollback khi lỗi.                                   |
| <b>5. Vận hành (Operation)</b>     | Giám sát, log, cập nhật phiên bản, scale hệ thống.                            | Quản lý sự cố, phân tích nguyên nhân lỗi, đảm bảo SLA.                        |

