**Họ và tên : Lê Thị Yên**

**MSSV: 20183861**

**Câu hỏi 1.**Vai trò của tầng Middleware trong các Hệ phân tán là gì?Lấy ví dụ về hoạt động của Middleware trong một hệ thống bất kỳ.

Trả lời:

* Vai trò của Middleware trong Hệ phân tán: giúp cải thiện tính trong suốt của hệ thống, giúp cho việc cấu hình là duy nhất

+ Hỗ trợ cài đặt ứng dụng phân tán

+ Che giấu sự không đồng nhất của keiesn trúc hệ thống bên dưới (phần cứng và phần mềm)

* Ví dụ:

Hệ thống cơ sở dữ liệu truyền thống thường được triển khai trong môi trường đóng, nơi người dùng chỉ truy cập vào hệ thống thông qua mạng bị hạn chế hoặc mạng nội bộ (ví dụ: mạng nội bộ của doanh nghiệp). Với sự phát triển phi thường của World Wide Web, người dùng có thể truy cập hầu như bất kỳ cơ sở dữ liệu nào mà họ có quyền truy cập thích hợp từ mọi nơi trên thế giới. Middleware giải quyết vấn đề về mức độ tương tác khác nhau giữa các cấu trúc cơ sở dữ liệu khác nhau. Phần mềm trung gian tạo điều kiện truy cập minh bạch vào các hệ thống quản lý cơ sở dữ liệu kế thừa (DBMS) hoặc các ứng dụng thông qua máy chủ web mà không cần quan tâm đến các đặc điểm cụ thể của cơ sở dữ liệu.

**Câu hỏi 2:** Thế nào là tính trong suốt của hệ thống? Có những kiểu trong suốt nào? Việc luôn để mức độ trong suốt của hệ thống lên mức cao nhất có phải là tốt không? Giải thích bằng một ví dụ.

Trả lời:

* Tính trong suốt là khả năng cung cấp một khung cảnh lôgic của hệ thống cho người dùng, độc lập với hạ tầng vật lý. Hệ thống luôn là duy nhất đối với người dùng song nó sẽ che giấu được tính phân tán của hệ phân tán phía dưới.

+ Giao diện giống nhau

+ Cách thức truy cập giống nhau

* Các kiểu trong suốt:
* Trong suốt truy cập (Access): Che giấu sự khác nhau trong biểu diễn dữ liệu và cách thức truy cập tài nguyên.
* Trong suốt vị trí (Location): Che giấu vị trí của tài nguyên, vị trí của TN không bị người sử dụng nhìn thấy.
* Trong suốt di trú (Migration): Che giấu việc tài nguyên chuyển đến địa điểm khác.
* Trong suốt về việc chuyển địa điểm (Relocation): Che giấu việc tài nguyên chuyển đến địa điểm khác ngay trong khi đang được sử dụng
* Trong suốt sao lưu (Replication): Che giấu việc dữ liệu được cung cấp từ nhiều bản sao khác nhau (thường được sử dụng rỗng rãi trong các hệ thống phân tán để tăng hiệu năng và tính sẵn sàng của hệ thống)
* Trong suốt tương tranh (Concurency): Che giấu việc tài nguyên được truy cập đồng thời bởi nhiều người sử dụng.
* Trong suốt sự cố (Failure): Che giấu lỗi và quá trình phục hồi của tài nguyên
* Trong suốt bền vững (Persistence): Che giấu việc tài nguyên/dữ liệu được lưu trữ bền vững (disk) hoặc không (RAM)

- Việc đảm bảo tính trong suốt là 1 trong những yêu cầu chắc chắn phải thực hiên để đảm bảo định nghĩa của hệ thống phân tán. Tuy nhiên để có được tính trong suốt ở mức độ tuyệt đối sẽ kéo theo chi phí về tài nguyên rất cao. Do đó không phải lúc nào cũng hướng tới trong suốt tuyệt đối

=> Cần xem xét trường hợp nào cần trong suốt đến đâu để tiết kiệm chi phí.

**Câu hỏi 3:** Nêu ưu điểm của kiến trúc Microservices so với kiến trúc Monolithic (đơn khối).

Trả lời:

**Ưu điểm** **của kiến trúc Microservices so với kiến trúc Monolithic:**

* Các component có kết nối lỏng lẻo dẫn đến dễ cách ly, dễ test và khởi động nhanh.
* Vòng đời phát triển nhanh hơn. Tính năng mới được phát triển nhanh hơn và tính năng cũ được cấu trúc lại dễ hơn.
* Các service có thể deploy độc lập nên ứng dụng dễ đọc, dễ tạo các bản vá hơn.
* Những issue, ví dụ liên quan đến memry leak một trong các service, bị cô lập và có thể không làm sập ứng dụng.
* Việc áp dụng các công nghệ mới dễ hơn. Các component có thể được nâng cấp độc lập với nhau.
* Các mô hình scale phức tạp và hiệu quả hơn có thể được thiết lập. Các service quan trọng có thể scale hiệu quả hơn. Các component riêng sẽ khởi động nhanh hơn và cải thiện thời gian khởi động của cả hệ thống.
* Các team tham gia sẽ ít phụ thuộc lẫn nhau. Kiến trúc này rất thích hợp cho các đội Agile.

**Câu hỏi 4:** Hãy thiết kế nhanh một hệ thống “nghe nhạc trực tuyến” theo kiến trúc Microservices. (gợi ý: vẽ ra các microservices và thông tin các services đó trao đổi với nhau là gì).

**Câu hỏi 5**: Trình bày ưu và nhược điểm của 3 loại kiến trúc server đa luồng sau: Thread-per-request (luồng với mỗi yêu cầu), Thread-per-connection (luồng với mỗi kết nối), Thread-per-object (luồng với mỗi đối tượng).

Trả lời:

|  |  |  |
| --- | --- | --- |
|  | Ưu điểm | Nhược điểm |
| Thread-per-request | * Không cần có hàng đợi vì mỗi yêu cầu gửi đến đều được sinh ra 1 luồng và xử lý ngay * Băng thông có thể đạt mức tối đa vì có khả năng sinh không giới hạn các luồng của worker | việc tạo, hủy luồng trong mỗi request nhận được, đặc biệt trong trường hợp có quá nhiều y/c gửi đến thì overhit của các thao tác hủy và tạo luồng sẽ rất cao làm giảm hiệu năng và làm tăng chi phí các hoạt động của hệ thống |
| Thread-per-connection | * Gắn kết mỗi luồng với 1 kết nối. Khi 1 client y/c kết nối đến với server, server sẽ tạo ra 1 luồng cho kết nối đó và hủy luồng nếu kết nối đó ngắt * Không tốn nhiều overhit vì không phải tạo và hủy nhiều luồng do số lượng đối tượng kết nối và đối tượng từ xa là không thay đổi nhiều | Các client có thể bị phục vụ với độ trễ cao do, trong khi 1 số luồng thì phục vụ quá nhiều y/c 1 số luồng khác thì không có việc để làm |
| Thread-per-object | * Tiến trình gắn 1 luồng với 1 đối tượng từ xa (remote object) mà mình quản lý. * Với kến trúc này luồng I/O phải quản lý thêm hàng đợi cho mỗi object * Không tốn nhiều overhit vì không phải tạo và hủy nhiều luồng do số lượng đối tượng kết nối và đối tượng từ xa là không thay đổi nhiều | Các client có thể bị phục vụ với độ trễ cao do, trong khi 1 số luồng thì phục vụ quá nhiều y/c 1 số luồng khác thì không có việc để làm |

**Câu hỏi 6:**Trình bày khái niệm ảo hóa. So sánh 2 mô hình kiến trúc ảo hóa là *Kiến trúc mô phòng hoàn toàn* và *Kiến trúc kiểm soát.*

Trả lời:

Ảo hóa giúpngười dùng dễ dàng quản trị các ứng dụng phần mềm

Diagram

Description automatically generated with medium confidence

* Hình a): Mô hình hoạt động của 1 hệ thống thông thường với việc sử dụng giao diện như 1 tầng trung gian giữa các chương trình ứng dụng và tầng dưới là hệ thống phần cứng A và phần mềm A. Tàng giao diện này cung cấp các phương thức được triển khai ở tầng hệ thống A bên dưới, tầng chương trình ở trên chỉ việc sử dụng giao diện này để gọi các phương thức đó
* Hình b): Khi đem cùng những chương trình kèm theo giao diện A sang 1 hệ thống khác (hệ thống B), lúc này cần 1 tầng trung gian để mô phỏng sự triển khai của hệ thống A trên hệ thống B. Cụ thể là triển khai các phương thức hệ thống A bằng các gọi phương thức tương ứng của hệ thống B mà giao diện B cung cấp. Tâng trung gian này mô phỏng ý tưởng của thực hiện ảo hóa (giả lập môi trường làm việc của hệ thống này trên môi trường làm việc của hệ thống khác)
* So sánh *Kiến trúc mô phòng hoàn toàn* và *Kiến trúc kiểm soát:*

**Câu hỏi 7:** Một hệ thống dự định triển khai giải pháp định danh phân cấp. Tuy nhiên, cứ mỗi lần thực thể chuyển động (thay đổi vị trí) mà lại cập nhật ngay vào hệ thống định danh phân cấp thì quá tốn kém. Hãy đề xuất giải pháp kết hợp với cơ chế *Chuyển tiếp con trỏ (Forwarding Pointers)* để tối ưu hơn hoạt động của hệ thống.

Giải pháp: Home-based Diagram

Description automatically generated

* Giải quyết vấn đề mở rộng, tính hiệu quả trong mạng quy mô lớn của broadcasting và forwarding pointers.
* Giải pháp này dựa trên việc cấp phát cho mỗi thực thể một vị trí gốc để lưu giữ địa chỉ hiện tại của các thực thể.
* Địa chỉ gốc được đăng kí tại dịch vụ đặt tên.

=> Như vậy để tìm kiếm một thực thể phải truy tìm vị trí gốc, để biết được chính xác vị trí hiện tại của một thực thể.