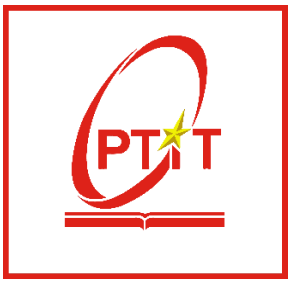
Học viện Công nghệ Bưu chính Viễn Thông



BÀI TẬP

MÔN CÁC HỆ THỐNG PHÂN TÁN NHÓM 2

Giảng viên: TS. Kim Ngọc Bách

Học viên: Hoàng Đức Cường - B24CHHT059

Phạm Đức Long - B24CHHT083

Nguyễn Đức Thức - B24CHHT094

Lớp học: M24CQHT02-B

Bắc Giang, tháng 7/2025

Câu 1: Phân tích hai đặc điểm quan trọng nhất của hệ thống phân tán và ba nguyên nhân khiến ứng dụng phân tán phức tạp hơn đơn lẻ.

Hệ thống phân tán (Distributed System) là một kiến trúc trong đó các thành phần được triển khai trên nhiều máy tính khác nhau nhưng hoạt động như một hệ thống duy nhất. Dưới đây là hai đặc điểm cốt lõi và ba yếu tố phức tạp hóa quá trình phát triển hệ thống phân tán:

Hai đặc điểm quan trọng của hệ thống phân tán

1. Tính tự trị và phân lập của các thành phần:
    - Mỗi node trong hệ thống phân tán là một thực thể độc lập, có bộ xử lý, bộ nhớ, hệ điều hành riêng và có thể thực hiện các tác vụ mà không cần phụ thuộc vào node khác.
    - Điều này tạo ra khả năng mở rộng linh hoạt và tăng độ sẵn sàng khi có sự cố cục bộ xảy ra.
2. Sự phối hợp để đạt được mục tiêu thống nhất:
    - Các thành phần của hệ thống phải phối hợp thông qua cơ chế truyền thông mạng để xử lý tác vụ chung.
    - Dữ liệu, trạng thái và tiến trình cần được đồng bộ hóa để người dùng có cảm giác sử dụng một hệ thống đơn nhất, không nhận ra sự phân tán về mặt vật lý.

Ba nguyên nhân khiến ứng dụng phân tán trở nên phức tạp hơn ứng dụng đơn lẻ

1. Môi trường mạng không ổn định:
    - Mạng máy tính luôn tiềm ẩn rủi ro như mất gói tin, độ trễ cao, lỗi kết nối hoặc tắc nghẽn.
    - Việc thiết kế các giao thức truyền thông cần đảm bảo tính tin cậy, thứ tự gói tin, khả năng retry và phát hiện lỗi.
2. Không đồng nhất về nền tảng phần cứng và phần mềm:
    - Các node có thể sử dụng kiến trúc CPU khác nhau (x86, ARM), hệ điều hành khác nhau (Linux, Windows) và ngôn ngữ lập trình khác nhau.
    - Yêu cầu sự tương thích và hỗ trợ bởi các chuẩn chung (như REST, gRPC, JSON, Protocol Buffers).
3. Yêu cầu cao về bảo mật và quản lý truy cập:
    - Dữ liệu di chuyển qua mạng và được lưu trữ ở nhiều nơi cần được mã hóa, xác thực và phân quyền chặt chẽ.

- Hệ thống cần triển khai các cơ chế bảo mật như TLS, RBAC (Role-Based Access Control), chứng thực người dùng và kiểm tra toàn vẹn dữ liệu.

Câu 2:

Phân tích các yếu tố phần cứng (CPU, bộ nhớ, kênh truyền) và yếu tố mạng (băng thông, topology) ảnh hưởng đến hiệu năng hệ thống phân tán.
Tại sao hệ điều hành phân tán (distributed OS) và hệ điều hành mạng (network OS) lại có yêu cầu khác nhau về quản lý tài nguyên?

Hiệu năng của hệ thống phân tán không chỉ phụ thuộc vào năng lực xử lý của từng node riêng lẻ, mà còn chịu ảnh hưởng đáng kể từ các yếu tố về kiến trúc phần cứng và môi trường mạng truyền thông. Một hệ thống được thiết kế tối ưu cần đảm bảo sự phối hợp nhịp nhàng giữa các yếu tố đó, cùng với một cơ chế điều phối tài nguyên hiệu quả từ hệ điều hành.

1. Ảnh hưởng của phần cứng và mạng đến hiệu năng hệ thống phân tán

a. CPU (Bộ xử lý trung tâm)

- CPU là thành phần then chốt quyết định tốc độ xử lý tác vụ tại mỗi node.
- Sự không đồng nhất về năng lực tính toán giữa các node dễ tạo ra "nút thắt cổ chai" (bottleneck), làm giảm hiệu suất chung của toàn hệ thống.
- Trong các ứng dụng HPC hoặc AI, sự phối hợp đồng bộ giữa CPU nhiều node còn yêu cầu các thuật toán phân chia tải tính toán (task partitioning) tối ưu.

b. Bộ nhớ (RAM)

- Trong hệ thống đơn lẻ, các tiến trình có thể chia sẻ bộ nhớ (shared memory). Tuy nhiên, trong hệ thống phân tán, mỗi node sử dụng bộ nhớ riêng biệt.
- Việc chia sẻ dữ liệu qua mạng làm tăng độ trễ, đòi hỏi chiến lược bộ nhớ tạm thời (caching) và tái sử dụng dữ liệu hợp lý để giảm truy xuất từ xa.

c. Kênh truyền nội bộ vs. Kết nối mạng

- Trong hệ thống đa xử lý (multiprocessor), truyền thông giữa các tiến trình nội bộ thường có độ trễ rất thấp (nanosecond).
- Trong khi đó, hệ thống phân tán phụ thuộc vào mạng LAN/WAN, nơi độ trễ có thể lên đến millisecond, chưa kể đến khả năng mất gói, nghẽn mạng, jitter.

- Các giao thức như TCP/IP được sử dụng, nhưng vẫn cần tối ưu hóa tầng ứng dụng bằng batch requests hoặc compressing payloads.

d. Băng thông mạng (Bandwidth)

- Băng thông quyết định tốc độ truyền dữ liệu giữa các node. Khi xử lý dữ liệu lớn (big data, media streaming), giới hạn băng thông có thể gây nghẽn.
- Các giải pháp bao gồm: data locality (xử lý gần nơi lưu trữ), sử dụng mạng tốc độ cao (InfiniBand), và cân bằng tải động (adaptive load balancing).

e. Topology mạng (Cấu trúc liên kết)

- Topology ảnh hưởng trực tiếp đến độ trễ và độ tin cậy của hệ thống.
- Ví dụ: mạng hình sao (star) đơn giản và hiệu quả trong mô hình client-server; mạng mesh hoặc hypercube phù hợp với HPC và các hệ thống cần độ sẵn sàng cao.
- Các thuật toán định tuyến (routing) và giao thức truyền thông (gossip, multicast) cần thiết kế phù hợp với topology cụ thể.

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

Tiêu chí	Hệ điều hành phân tán (Distributed OS)	Hệ điều hành mạng (Network OS)
Mục tiêu chính	Cung cấp trải nghiệm "như một hệ điều hành duy nhất"	Cho phép chia sẻ tài nguyên giữa các máy
Quản lý tài nguyên	Tập trung, trong suốt, toàn hệ thống	Cục bộ, không trong suốt
Giao tiếp	Transparent RPC, message passing	File sharing, remote login (e.g., FTP, Telnet)
Ví dụ thực tế	Amoeba OS, Google Borg, Plan 9	Windows Server, UNIX với NFS

Phân tích sâu:

- Distributed OS yêu cầu lớp điều phối tài nguyên toàn cục (global resource manager) để lên lịch (scheduling), giám sát trạng thái (monitoring), và cấp phát tài nguyên động.
- Network OS không cần điều phối thống nhất. Mỗi máy chủ tự quản lý tài nguyên, chỉ chia sẻ theo yêu cầu từ client.
- Distributed OS cần mô hình đồng bộ hóa mạnh (strong consistency) hơn do tính chất phụ thuộc lẫn nhau giữa các node.

Kết luận, một hệ thống phân tán hiệu quả phải được xây dựng trên cơ sở phần cứng đồng đều, kiến trúc mạng ổn định và thiết kế hệ điều hành phù hợp với mục tiêu sử dụng. Distributed OS là một bước tiến cao hơn của Network OS, nhắm tới khả năng phối hợp tổng thể và che giấu tính phân tán để đơn giản hóa trải nghiệm cho người dùng và lập trình viên.

Câu 3:

Nêu và so sánh ba loại hệ thống phân tán: điện toán phân tán, thông tin phân tán và lan tỏa phân tán.
Phân tích các lớp chính (application, middleware, resource) trong kiến trúc điện toán lưới và vai trò của từng lớp.

Trong hệ sinh thái hệ thống phân tán hiện đại, nhu cầu xử lý thông tin và tính toán theo các mô hình đa dạng đã dẫn đến sự phân hóa thành ba loại hệ thống phân tán chính: điện toán phân tán, thông tin phân tán và lan tỏa phân tán. Mỗi loại phục vụ cho một mục tiêu khác nhau và có kiến trúc triển khai đặc thù.

1. So sánh ba loại hệ thống phân tán

Tiêu chí	Điện toán phân tán	Thông tin phân tán	Lan tỏa phân tán
Mục tiêu chính	Tối ưu hiệu suất xử lý tính toán	Phân tán lưu trữ và truy cập dữ liệu	Tích hợp thiết bị và cảm biến vào môi trường sống
Đặc điểm chính	Chia nhỏ tác vụ lớn để xử lý song song trên nhiều node	Lưu trữ và truy xuất dữ liệu từ nhiều vị trí vật lý khác nhau	Thiết bị nhúng, cảm biến, hoạt động tự động không can thiệp người dùng
Cách triển khai	Cụm máy chủ (cluster), điện toán lưới (grid), siêu máy tính	Hệ thống CDN, cơ sở dữ liệu phân tán, blockchain	IoT, thiết bị thông minh, pervasive computing
Ví dụ	Tính toán mô phỏng thời tiết, AI training	Google Bigtable, Amazon DynamoDB	Smart home, hệ thống đo môi trường tự động

Phân tích :

- Điện toán phân tán (Distributed Computing): Tập trung giải quyết các bài toán nặng về xử lý, yêu cầu đồng bộ cao, sử dụng cơ chế chia nhỏ task và tập hợp kết quả. Thường được áp dụng trong môi trường hiệu năng cao (HPC).
- Thông tin phân tán (Distributed Information): Phục vụ lưu trữ và truy vấn dữ liệu ở quy mô lớn. Cân nhắc tính nhất quán dữ liệu (consistency) và độ trễ truy cập (latency) là yếu tố then chốt.
- Lan tỏa phân tán (Pervasive/Ubiquitous Computing): Các node nhỏ, tiêu thụ năng lượng thấp, gắn vào đời sống hàng ngày, hoạt động dựa trên sự kiện (event-driven). Thường dùng trong smart city, smart healthcare.

2. Kiến trúc điện toán lưới và vai trò của các lớp

Kiến trúc điện toán lưới (Grid Computing Architecture) được xây dựng theo hướng phân tầng, trong đó mỗi tầng đảm nhận một lớp chức năng riêng biệt từ người dùng đến tầng tài nguyên vật lý.

a. Lớp ứng dụng (Application Layer)

- Cung cấp giao diện tương tác cho người dùng cuối.
- Chịu trách nhiệm chuyển yêu cầu người dùng thành các tác vụ tính toán phân tán.
- Thường tích hợp các cổng dịch vụ (service portals), hoặc giao diện đồ họa (GUI).

b. Lớp phần mềm trung gian (Middleware Layer)

- Là trung tâm điều phối toàn bộ tài nguyên lưới.
- Cung cấp các chức năng chính như:
    - Quản lý tài nguyên (Resource Management): phát hiện, theo dõi, cấp phát tài nguyên.
    - Lên lịch tác vụ (Job Scheduling): xác định vị trí phù hợp để chạy từng task.
    - Bảo mật và xác thực (Security & Authentication): sử dụng certificate, token để bảo vệ truy cập.
- Đây là lớp hiện thực các tiêu chuẩn dịch vụ lưới (OGSA - Open Grid Services Architecture).

c. Lớp tài nguyên (Resource Layer)

- Là lớp giao tiếp với phần cứng vật lý: máy chủ, bộ nhớ, ổ lưu trữ, mạng.

- Cung cấp thông tin trạng thái tài nguyên, khả năng truy xuất và cơ chế cấp phát dưới sự điều phối của middleware.
- Cần đảm bảo tính ổn định và tương thích với phần mềm trung gian qua các trình điều khiển (drivers, resource adapters).

Tổng kết, việc lựa chọn loại hệ thống phân tán và kiến trúc phù hợp cần dựa trên mục tiêu ứng dụng cụ thể: nếu yêu cầu tính toán phức tạp, điện toán lưới là giải pháp; nếu ưu tiên lưu trữ phân tán và khả năng mở rộng, cần hướng đến kiến trúc dữ liệu phân tán; còn nếu tích hợp sâu với môi trường vật lý thì hệ thống lan tỏa là lựa chọn tối ưu.

Câu 4:

Giải thích tại sao "tính sẵn sàng" (availability) được xem là mục tiêu quan trọng nhất của hệ thống phân tán.
Nêu và so sánh ba hình thức "tính trong suốt" (trong suốt về truy nhập, vị trí và lỗi), và ví dụ đơn giản minh họa mỗi loại.
Trình bày mối quan hệ giữa "tính mở" (openness) và khả năng tương tác (interoperability) trong hệ thống phân tán.

1. Tính sẵn sàng (Availability) – Mục tiêu sống còn của hệ thống phân tán

Tính sẵn sàng thể hiện khả năng hệ thống cung cấp dịch vụ liên tục cho người dùng, bất kể các lỗi cục bộ hoặc sự cố về hạ tầng. Trong bối cảnh phân tán, khi các tài nguyên và dịch vụ được triển khai trên nhiều node, khả năng một thành phần bị lỗi là không thể tránh khỏi. Do đó, tính sẵn sàng là chỉ tiêu hàng đầu trong thiết kế hệ thống.

Vì sao tính sẵn sàng là ưu tiên hàng đầu:

- Trực tiếp liên quan đến trải nghiệm người dùng: Một hệ thống thương mại điện tử hay ngân hàng số không thể chấp nhận downtime nếu không muốn mất uy tín và doanh thu.
- Liên kết với khả năng chịu lỗi (fault tolerance): Tính sẵn sàng đạt được nhờ các cơ chế như replication, failover, auto-recovery.
- Gắn liền với các chỉ số SLA (Service Level Agreement): Nhiều tổ chức cam kết duy trì uptime > 99.99%, tương đương chưa đầy 1 giờ gián đoạn mỗi năm.

Ví dụ: Hệ thống dịch vụ của Amazon Web Services áp dụng mô hình multi-region để đảm bảo nếu một khu vực bị sự cố, các dịch vụ vẫn được duy trì ở các khu vực khác.

2. Ba dạng "tính trong suốt" và ý nghĩa trong hệ thống phân tán

"Tính trong suốt" giúp che giấu sự phức tạp của hệ thống phân tán khỏi người dùng cuối, đồng thời đơn giản hóa lập trình và bảo trì hệ thống.

Loại trong suốt	Mô tả	Ví dụ thực tiễn
Truy nhập (Access Transparency)	Người dùng có thể truy cập tài nguyên mà không cần quan tâm đến cách thức hay định dạng truy cập	Truy cập tệp qua WebDAV hoặc NFS giống như truy cập tệp cục bộ
Vị trí (Location Transparency)	Tài nguyên có thể được truy cập mà không cần biết vị trí vật lý của nó	Gọi API tại `api.service.com` mà không biết server đặt tại Singapore hay Mỹ
Lỗi (Failure Transparency)	Hệ thống tự phục hồi khi có lỗi mà người dùng không nhận ra	Khi server chính bị lỗi, hệ thống tự động chuyển sang server backup

Phân tích:

- Việc hiện thực tính trong suốt cần middleware có khả năng định tuyến, chuyển tiếp, phát hiện lỗi và khôi phục.
- Quản trị viên có thể cấu hình lại hệ thống, thay đổi topology hoặc thay thế tài nguyên mà không làm gián đoạn dịch vụ.

3. Mối quan hệ giữa tính mở (Openness) và khả năng tương tác (Interoperability)

Tính mở (Openness)

- Hệ thống mở cho phép tích hợp các thành phần từ nhiều nhà phát triển, sử dụng các công nghệ và chuẩn giao tiếp đa dạng.
- Hệ thống mở thường dựa trên các chuẩn công nghiệp: HTTP, REST, SOAP, XML, JSON, OAuth,...

Khả năng tương tác (Interoperability)

- Là khả năng các hệ thống hoặc thành phần khác nhau có thể làm việc cùng nhau một cách hiệu quả.
- Đạt được nhờ có giao diện rõ ràng, chuẩn hóa và không phụ thuộc nền tảng.

Mối quan hệ:

- Tính mở là điều kiện cần để đạt được khả năng tương tác.
- Một hệ thống càng mở, càng dễ dàng tích hợp với các dịch vụ bên ngoài, nâng cao khả năng mở rộng và đổi mới công nghệ.

Ví dụ: Một hệ thống thanh toán điện tử tích hợp với nhiều ví điện tử (Momo, ZaloPay, VNPay) phải tuân thủ các chuẩn API và giao thức xác thực chung để duy trì tính tương tác.

Tóm lại, tính sẵn sàng, tính trong suốt và tính mở là ba trụ cột quan trọng để xây dựng một hệ thống phân tán hiện đại, hiệu quả và dễ bảo trì. Thiếu một trong ba yếu tố này sẽ làm suy giảm đáng kể giá trị sử dụng và khả năng vận hành lâu dài của hệ thống.

Câu 5

So sánh ưu – nhược điểm của kiến trúc phân cấp và kiến trúc ngang hàng trong hệ thống phân tán.
Trình bày bốn mô hình hệ thống phân tán (phân tầng, đối tượng phân tán, kênh sự kiện, dữ liệu tập trung) và cho ví dụ ứng dụng điển hình cho mỗi mô hình.

Nêu vai trò của phần mềm trung gian (middleware) trong kiến trúc khách-chủ phân tán, và liệt kê ba tính năng chính mà nó cung cấp

1. So sánh kiến trúc phân cấp và kiến trúc ngang hàng trong hệ thống phân tán

Hai kiến trúc cơ bản nhất trong các hệ thống phân tán là kiến trúc phân cấp (hierarchical) và kiến trúc ngang hàng (peer-to-peer). Mỗi kiến trúc phù hợp với một kiểu ứng dụng và mang theo các ưu – nhược điểm riêng biệt.

Tiêu chí	Kiến trúc phân cấp	Kiến trúc ngang hàng
Cấu trúc	Hình cây: nút cha điều phối các nút con	Các nút có vai trò tương đương, không có trung tâm
Ưu điểm	Dễ kiểm soát, quản lý tập trung, phân quyền rõ ràng	Tăng khả năng chịu lỗi, phân tán tải, dễ mở rộng ngang
Nhược điểm	Điểm nghẽn tại nút trung tâm; dễ bị ảnh hưởng nếu nút điều phối bị lỗi	Đồng thuận phức tạp; khó kiểm soát hành vi đồng bộ

Ứng dụng	DNS, LDAP, hệ thống file tập trung	BitTorrent, Blockchain, Hệ thống lưu trữ phân tán

Phân tích:

- Kiến trúc phân cấp thường phù hợp với hệ thống tổ chức rõ ràng, ví dụ các hệ thống chính phủ, ngân hàng.
- Kiến trúc ngang hàng phát huy hiệu quả trong môi trường cần mở rộng động, chịu lỗi mạnh như mạng chia sẻ tệp, ứng dụng blockchain.

2. Bốn mô hình hệ thống phân tán và ví dụ

Mô hình	Đặc điểm	Ứng dụng điển hình
Phân tầng (Layered Model)	Các tầng thực hiện chức năng riêng biệt, giao tiếp qua interface chuẩn	Kiến trúc Web 3-tier: frontend, backend, database
Đối tượng phân tán (Distributed Object)	Mỗi đối tượng phân bố trên các node, giao tiếp thông qua gọi thủ tục từ xa (RMI, CORBA)	Hệ thống thương mại điện tử đa nền tảng
Kênh sự kiện (Event-based Model)	Thành phần trung gian tiếp nhận, phân phối sự kiện giữa các producer-consumer	Apache Kafka, MQTT trong IoT
Dữ liệu tập trung (Centralized Data)	Dữ liệu tập trung tại server trung tâm, client truy cập qua mạng	Hệ thống quản lý sinh viên sử dụng cơ sở dữ liệu tập trung

Phân tích:

- Các mô hình này có thể được kết hợp để thiết kế hệ thống thực tế. Ví dụ: một ứng dụng thương mại điện tử có thể sử dụng mô hình phân tầng cho frontend/backend, đồng thời tích hợp kênh sự kiện để xử lý thông báo thời gian thực.

3. Vai trò và tính năng chính của phần mềm trung gian (Middleware)

Phần mềm trung gian đóng vai trò là tầng trung gian kết nối giữa tầng ứng dụng và hạ tầng truyền thông. Đây là yếu tố cốt lõi để hiện thực hệ thống phân tán theo hướng module hóa, che giấu sự phân tán khỏi người dùng và nhà phát triển.

Vai trò chính:

- Tạo lớp trừu tượng cho phân tán: Middleware giúp ứng dụng tương tác với tài nguyên từ xa giống như đang làm việc cục bộ.
- Cung cấp các dịch vụ cơ bản: Đồng bộ, bảo mật, định danh, phát hiện lỗi, và quản lý phiên làm việc.
- Đảm bảo khả năng mở rộng và khả năng tương tác: Hỗ trợ nhiều giao thức, API và chuẩn công nghiệp.

Ba tính năng nổi bật:

1. Che giấu tính phân tán (Transparency): Người dùng không cần biết vị trí tài nguyên hay node đang xử lý.
2. Cung cấp dịch vụ phổ biến (Common Services): Định tuyến thông điệp, điều phối tiến trình, chuyển đổi định dạng dữ liệu.
3. Hỗ trợ giao tiếp dị nền tảng: Cho phép hệ thống sử dụng ngôn ngữ, hệ điều hành khác nhau tương tác trơn tru (ví dụ: REST API, gRPC).

Ví dụ: Hệ thống ngân hàng điện tử có thể sử dụng middleware như Apache Camel để tích hợp giữa các dịch vụ thanh toán, xác thực người dùng, và hệ thống lưu trữ dữ liệu.

Tóm lại, việc lựa chọn kiến trúc và mô hình phân tán phù hợp cùng với vai trò thiết yếu của middleware là yếu tố quyết định sự thành công khi xây dựng một hệ thống phân tán hiệu quả, có khả năng mở rộng, bảo trì tốt và phục vụ lâu dài.

Câu 6:

Phân loại ba loại dịch vụ trong SOA (cơ bản, tích hợp, quy trình) kèm ví dụ điển hình cho mỗi loại.

Trình bày vòng đời của một dịch vụ SOA, từ giai đoạn phát triển đến vận hành sản xuất, và những thách thức chính ở mỗi giai đoạn.

1. Phân loại ba loại dịch vụ trong SOA (Service-Oriented Architecture)

Kiến trúc hướng dịch vụ (SOA) là một phong cách thiết kế trong đó các thành phần phần mềm cung cấp chức năng dưới dạng dịch vụ được truy cập từ xa và độc lập với nền tảng. Trong SOA, dịch vụ có thể được phân loại thành ba nhóm chính:

Loại dịch vụ	Mô tả	Ví dụ

Dịch vụ cơ bản (Basic Services)	Cung cấp chức năng nghiệp vụ đơn lẻ, thường là các tác vụ nguyên tử	Dịch vụ kiểm tra tồn kho, dịch vụ tra cứu sản phẩm
Dịch vụ tích hợp (Composite Services)	Tổ hợp nhiều dịch vụ cơ bản để hoàn thành một tiến trình phức tạp hơn	Dịch vụ đặt hàng gồm kiểm tra tồn kho, tính giá, thanh toán
Dịch vụ quy trình (Process Services)	Mô hình hóa toàn bộ quy trình nghiệp vụ, bao gồm logic điều hướng, thứ tự thực hiện, ràng buộc	Quy trình xử lý yêu cầu hoàn tiền, quản lý chuỗi cung ứng

Phân tích:

- Dịch vụ cơ bản mang tính tái sử dụng cao và được xem như "building blocks" trong hệ thống.
- Dịch vụ tích hợp là lớp trung gian, nơi điều phối các logic dịch vụ nhỏ thành khối chức năng hoàn chỉnh.
- Dịch vụ quy trình là tầng cao nhất, cần công cụ điều phối (orchestration engine) như BPEL, BPMN để mô tả và thực thi.

2. Vòng đời của dịch vụ SOA và các thách thức theo từng giai đoạn

Vòng đời phát triển và triển khai một dịch vụ SOA có thể chia thành bốn giai đoạn chính:

Giai đoạn	Mô tả	Thách thức chính
Phân tích & Thiết kế	Xác định nghiệp vụ cần số hóa, xác lập giao diện và mô hình dữ liệu dịch vụ	Yêu cầu không rõ ràng, khó mô hình hóa quá trình kinh doanh
Phát triển	Cài đặt chức năng dịch vụ, kiểm thử đơn vị và tích hợp	Đảm bảo tuân thủ tiêu chuẩn (WSDL, REST), xử lý tương thích với hệ thống cũ
Triển khai	Đăng ký dịch vụ lên service registry, cấu hình endpoint, chính sách bảo mật	Vấn đề bảo mật, tối ưu hạ tầng, kiểm soát versioning
Vận hành & Bảo trì	Giám sát, ghi log, kiểm soát hiệu suất và thay đổi	Đảm bảo uptime, tương thích ngược, quản lý thay đổi giao diện