HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CÁC HỆ THỐNG PHÂN TÁN

BÀI TẬP SESSION/CHAPTER 1

Giảng viên:	TS. Kim Ngọc Bách	
Lớp:	M24CQHT02-B	
Thực hiện:	Nhóm 12	
	Trần Đức Tiến Long	B24CHHT082
	Đào Văn Luân	B24CHHT084
	Trần Xuân Huỳnh	B24CHHT079

Hà Nội - 2025

Câu 1:

Nêu và giải thích hai đặc điểm quan trọng nhất của hệ thống phân tán.
Trình bày ba lý do cơ bản khiến các ứng dụng phân tán phức tạp hơn so với các ứng dụng đơn lẻ.

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

    1. Tập hợp các phần tử tính toán tự trị (autonomous computing elements): mỗi nút (node) trong hệ thống phân tán có thể là một thiết bị phần cứng hoặc một tiến trình phần mềm độc lập, vận hành theo đồng hồ riêng và có thể gia nhập/rời nhóm bất cứ lúc nào. Sự tự trị này dẫn đến các thách thức về đồng bộ và điều phối giữa các nút, vì không có "đồng hồ chung" và mỗi nút phải tự quản lý thành viên cũng như chứng thực lẫn nhau .

    2. Hiện ra như một hệ thống gắn kết (single coherent system): người dùng hoặc ứng dụng nhìn thấy toàn bộ mạng phân tán như một thực thể duy nhất. Điều này đòi hỏi phải che giấu việc dữ liệu hay tính toán được phân bố trên nhiều nút (distribution transparency), đồng thời phải ứng phó với thất bại cục bộ (partial failures), vì một nút hỏng không được để làm gián đoạn toàn hệ thống .

2. Ba lý do cơ bản khiến ứng dụng phân tán phức tạp hơn so với ứng dụng đơn lẻ

    1. Chia sẻ tài nguyên và dữ liệu: người dùng có thể truy cập dữ liệu từ xa mà không sao chép cục bộ, dẫn đến yêu cầu về phân quyền, xác thực, và bảo mật phức tạp hơn .

    2. Hiệu năng và độ trễ mạng: giao tiếp qua mạng luôn có độ trễ (từ hàng chục đến hàng trăm ms), băng thông không vô hạn, buộc các ứng dụng phải thiết kế song song, bất đồng bộ hoá, caching và nhân bản dữ liệu để giảm bottleneck .

    3. Khả năng chịu lỗi (fault tolerance): trong môi trường phân tán, bất kỳ nút hay liên kết nào cũng có thể hỏng bất thình lình; hệ thống phải phát hiện và chuyển tải công việc sang nút khác để đảm bảo dịch vụ không bị gián đoạn .

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?

1. Ảnh hưởng của yếu tố phần cứng đến hiệu năng

    - CPU: tốc độ xung nhịp và số lõi quyết định khả năng xử lý song song; tuy nhiên, khi số lõi tăng, chi phí điều phối giữa các lõi (cache coherence, bus contention) càng cao .

    - Bộ nhớ: lựa chọn kiến trúc bộ nhớ đồng nhất (UMA) hay không đồng nhất (NUMA) ảnh hưởng trực tiếp đến độ trễ đọc/ghi và khả năng mở rộng; NUMA dùng cache cục bộ để giảm tắc nghẽn kênh truyền chung .

    - Kênh truyền nội bộ (on-board interconnect): bus chung dễ bị "nghẽn cổ chai" khi nhiều CPU truy xuất đồng thời; chuyển mạch (switch) hoặc kết nối từng cặp (point-to-point) giúp tăng băng thông nội bộ.

2. Ảnh hưởng của yếu tố mạng đến hiệu năng

    - Băng thông (bandwidth): giới hạn tốc độ truyền khối lượng lớn dữ liệu; ứng dụng phân tán cần tối ưu hoá kích thước gói tin, batching, và compression để tận dụng tối đa băng thông .

    - Topologies (hình dạng mạng): mạng bus dễ dẫn đến xung đột chia sẻ kênh; mạng hình sao với switch sẽ phân vùng xung đột theo cổng, cho phép mỗi liên kết chạy độc lập; mạng diện rộng (WAN) thường có độ trễ cao hơn nhiều so với LAN, đòi hỏi thiết kế giao thức có khả năng chịu trễ và mất gói.

3. Khác biệt về quản lý tài nguyên giữa Distributed OS và Network OS

    - Distributed OS (HĐH phân tán): che giấu hoàn toàn tính phân tán, cung cấp giao diện thống nhất với tính trong suốt cao (process migration, unified file system), do đó cần cơ chế đồng bộ phân tán, quản lý tham chiếu tài nguyên toàn cục và chịu lỗi mạnh mẽ.

- Network OS (HĐH mạng): chỉ cung cấp dịch vụ chia sẻ file/printer, user authentication, không che giấu hoàn toàn phân tán; tài nguyên vẫn được quản lý cục bộ, ứng dụng phải biết rõ vị trí tài nguyên.

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.

1. Ba loại hệ thống phân tán

   - Điện toán phân tán (Distributed Computing): Tập trung nâng cao khả năng xử lý thông qua phân phối công việc tính toán lên nhiều nút (cluster, grid). Mục tiêu chính là tối ưu hiệu năng và tận dụng tài nguyên tính toán dư thừa trên nhiều máy .

   - Thông tin phân tán (Distributed Information): Tập trung quản lý và truy xuất dữ liệu được chia sẻ, sao chép hoặc phân mảnh trên nhiều nút. Mục đích là đảm bảo tính nhất quán, khả năng mở rộng và độ bền của dữ liệu .

   - Lan tỏa phân tán (Pervasive/Disseminated): Nhấn mạnh vào việc phân phối thông tin tự động tới người dùng hoặc thiết bị cuối (ví dụ IoT, sensor networks), chú trọng tính linh động, di động và tác động tới người dùng cá nhân .

2. So sánh chung:

Tiêu chí	Điện toán phân tán	Thông tin phân tán	Lan tỏa phân tán
Mục tiêu chính	Hiệu năng tính toán	Quản lý & nhất quán dữ liệu	Phân phối thông tin tức thời
Tài nguyên	CPU, bộ nhớ, GPU…	Hệ quản trị cơ sở dữ liệu, cache…	Thiết bị nhúng, cảm biến…
Ví dụ điển hình	Grid computing (BOINC, HTCondor)	Hệ quản trị CSDL phân tán (Cassandra)	Mạng cảm biến không dây

3.

Kiến trúc điện toán lưới (Grid Computing)

– Lớp tài nguyên (Resource Layer): Cung cấp giao diện truy cập đến tài nguyên tính toán, lưu trữ, thiết bị chuyên dụng… cho tầng trên thông qua các API chuẩn .

– Lớp trung gian (Middleware Layer): Đóng vai trò "kết nối" giữa ứng dụng và tài nguyên, đảm nhận chức năng định vị, phân phối công việc, cân bằng tải, bảo mật, giao thức truyền thông và quản lý dịch vụ .

– Lớp ứng dụng (Application Layer): Bao gồm các phần mềm người dùng hoặc dịch vụ chạy trên lưới, tận dụng các dịch vụ middleware để triển khai và điều phối công việc phân tán.

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. Tại sao "tính sẵn sàng" (availability) là mục tiêu quan trọng nhất?
   Trong môi trường phân tán, thất bại cục bộ không thể tránh khỏi (mạng chậm, nút chết…), nhưng người dùng và ứng dụng vẫn cần truy cập tài nguyên mọi lúc. Nếu không đảm bảo tính sẵn sàng, dịch vụ sẽ gián đoạn và mất tin cậy .

2. Ba hình thức tính trong suốt và ví dụ minh họa

   - Trong suốt về truy nhập (Access Transparency): Che giấu cách thức và định dạng truy cập tài nguyên. Ví dụ: người dùng truy cập file bằng đường dẫn chung bất kể filesystem thực sự là NTFS, EXT4 hay mạng .

   - Trong suốt về vị trí (Location Transparency): Che giấu vị trí vật lý của tài nguyên. Ví dụ: URL "www.ptit.edu.vn" không tiết lộ địa chỉ hoặc máy chủ cụ thể lưu trang .

   - Trong suốt về lỗi (Failure Transparency): Che giấu sự cố và tự khôi phục trước khi thông báo cho người dùng. Ví dụ: middleware tự đánh giá lỗi nút, chuyển công việc sang nút khác và trả kết quả bình thường .

3. Tính mở (openness) và khả năng tương tác (interoperability)
   – Tính mở: Hệ thống tuân thủ các chuẩn giao tiếp và mô tả dịch vụ trung lập, dễ dàng mở rộng hoặc thay thế thành phần mà không ảnh hưởng các phần khác .
   – Khả năng tương tác: Mức độ mà các thành phần từ nhà cung cấp khác nhau có thể làm việc chung dựa trên các giao diện chuẩn. Được đảm bảo bởi tính mở (định nghĩa giao diện/API trung lập), tạo điều kiện cho tính tương tác cao .

   Mối quan hệ: Openness tạo nền tảng (tiêu chuẩn, mô-đun hóa) để các thành phần có thể tương tác dễ dàng; interoperability là kết quả trực tiếp khi hệ thống đủ mở.

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. Ưu – nhược điểm kiến trúc phân cấp vs ngang hàng
   - Phân cấp (Hierarchical):
     1. Ưu điểm: Dễ quản lý, kiểm soát truy cập, mở rộng theo tầng, phân vùng rõ ràng .
     2. Nhược điểm: Núi cha hỏng thì cả nhánh chịu ảnh hưởng, có điểm nghẽn cổ chai tại các nút trung tâm .
   - Ngang hàng (Peer-to-Peer):
     1. Ưu điểm: Không có điểm tập trung, khả năng chịu lỗi cao, phân phối cân bằng tải; mỗi nút vừa là client vừa là server .
     2. Nhược điểm: Khó kiểm soát, đòi hỏi cơ chế định vị/phát hiện nút phức tạp, bảo mật và đồng bộ dữ liệu thách thức hơn.

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

    - Phân tầng (Layered): Các tầng rõ ràng (presentation, logic, data); ví dụ kiến trúc web MVC.

    - Đối tượng phân tán (Distributed Object): Các đối tượng được gọi từ xa; ví dụ CORBA, Java RMI.

    - Kênh sự kiện (Event-based/Message Channel): Pub/Sub, event bus; ví dụ Apache Kafka, MQTT.

    - Dữ liệu tập trung (Shared Data): Client-server với CSDL tập trung; ví dụ hệ thống DNS, LDAP.

3. Vai trò của middleware trong kiến trúc khách-chủ phân tán và ba tính năng chính

    - Vai trò: Đứng giữa client và server, che giấu chi tiết truyền thông, cung cấp API đồng nhất, xử lý marshalling/unmarshalling, naming, bảo mật, giao dịch…

    - Ba tính năng chính:

        1. Định danh & tìm kiếm tài nguyên (naming/directory services)

        2. Điều phối giao tiếp & marshalling (RPC, message queuing)

        3. Quản lý giao dịch & đồng bộ (transactions, concurrency control)

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. Ba loại dịch vụ trong SOA & ví dụ

    - Dịch vụ cơ bản (Basic/Atomic Service): Thao tác nhỏ, không trạng thái, thời gian thực hiện ngắn. Ví dụ: “CreateUser”, “GetAccountBalance” .

- Dịch vụ tích hợp (Integration Service): Kết hợp nhiều dịch vụ cơ bản, vẫn không trạng thái, có thể跨 nền tảng. Ví dụ: “VerifyCustomerAndInitiateLoan” .

- Dịch vụ qui trình (Process Service): Mô tả quy trình nghiệp vụ, có trạng thái, thời gian thực hiện dài. Ví dụ: “LoanApprovalProcess” với nhiều bước (tạo yêu cầu, thẩm định, phê duyệt…) .

2. Vòng đời dịch vụ SOA & thách thức

    - Phát triển (Design & Build): Xác định giao diện (WSDL/IDL), thiết kế hợp đồng, mô hình hóa quy trình. *Thách thức:* Định nghĩa ngữ nghĩa giao diện đầy đủ, đảm bảo tính mô-đun và đóng gói cao .

    - Triển khai & Kiểm thử (Deploy & Test): Cấu hình ESB/Enterprise Service Bus, thiết lập routing, bảo mật, QoS. *Thách thức:* Đồng bộ môi trường, kiểm thử tích hợp đa nền tảng.

    - Vận hành (Production & Monitor): Giám sát SLAs, logging, versioning, quản lý thay đổi. *Thách thức:* Duy trì tính tương thích ngược, nâng cấp không gián đoạn, đảm bảo performance và bảo mật liên tục.