NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

CHƯƠNG 4: THIẾT KẾ HỆ THỐNG

Giảng viên: TS. Đỗ Thị Thanh Tuyền
Email: tuyendtt@uit.edu.vn

NỘI DUNG

I. Kiến trúc hệ thống

II. Một số kiến trúc hệ thống

I. Kiến trúc hệ thống

Khái niệm:

Hệ thống được cấu tạo bởi các thành phần nào và mối liên hệ giữa các thành phần đó.

I. Kiến trúc hệ thống (tt)

Tầm quan trọng của kiến trúc:

- Ảnh hưởng hiệu quả hoạt động và an toàn hệ thống:
  - + Tốc độ xử lý
  - + Tính chịu lỗi
  - + Tính bảo mật
- Ảnh hưởng chi phí triển khai, vận hành và bảo trì hệ thống.
- Ảnh hưởng trực tiếp đến giai đoạn thiết kế.

I. Kiến trúc hệ thống (tt)

Các bước thiết kế kiến trúc:

- Phân rã hệ thống -> xác định các thành phần;
- Bố trí các thành phần;
- Thiết lập mối quan hệ giữa các thành phần.

II. Một số kiến trúc hệ thống

1. **Kiến trúc đơn lập**
2. **Kiến trúc phân tán:**
    - Kiến trúc 2 lớp
    - Kiến trúc 3 lớp
    - Kiến trúc đa tầng
    - Kiến trúc Peer-to-Peer

II.1 Kiến trúc đơn lập

- **Đặc điểm:**
  - Là một thể thống nhất.
  - Các thành phần tự do tương tác.
- **Ưu điểm:**
  - Dễ lập trình và triển khai.
  - Tốc độ xử lý.
- **Khuyết điểm:**
  - Khó bảo trì, nâng cấp.
  - Không chia sẻ dữ liệu.

II.2 Kiến trúc phân tán

- **Kiến trúc 2 lớp (Client – Server)**
- **Kiến trúc 3 lớp**
- **Kiến trúc đa tầng**
- **Kiến trúc Peer–to–Peer (P2P)**

II.2.1 Kiến trúc Client – Server

- **Đặc điểm:**

  - Phân làm hai phân hệ Client và Server.

  - Server cung cấp dịch vụ, Client sử dụng dịch vụ.

  - Client gửi các yêu cầu đến Server.

- **Ưu điểm:**

  - Chia sẻ dữ liệu và đồng bộ.

  - Dễ bảo trì, nâng cấp.

- **Khuyết điểm:**

  - Tốc độ xử lý.

  - Chi phí triển khai.

II.2.1 Kiến trúc Client – Server (tt)

- **Thick Server (Thin-Client)**

  - Server = Dữ liệu + **Xử lý**

  - Client = Giao diện

- **Thick Client (Thin-Server)**

  - Server = Dữ liệu

  - Client = Giao diện + **Xử lý**

Ví dụ Kiến trúc Client-Server

1) SQL Server:

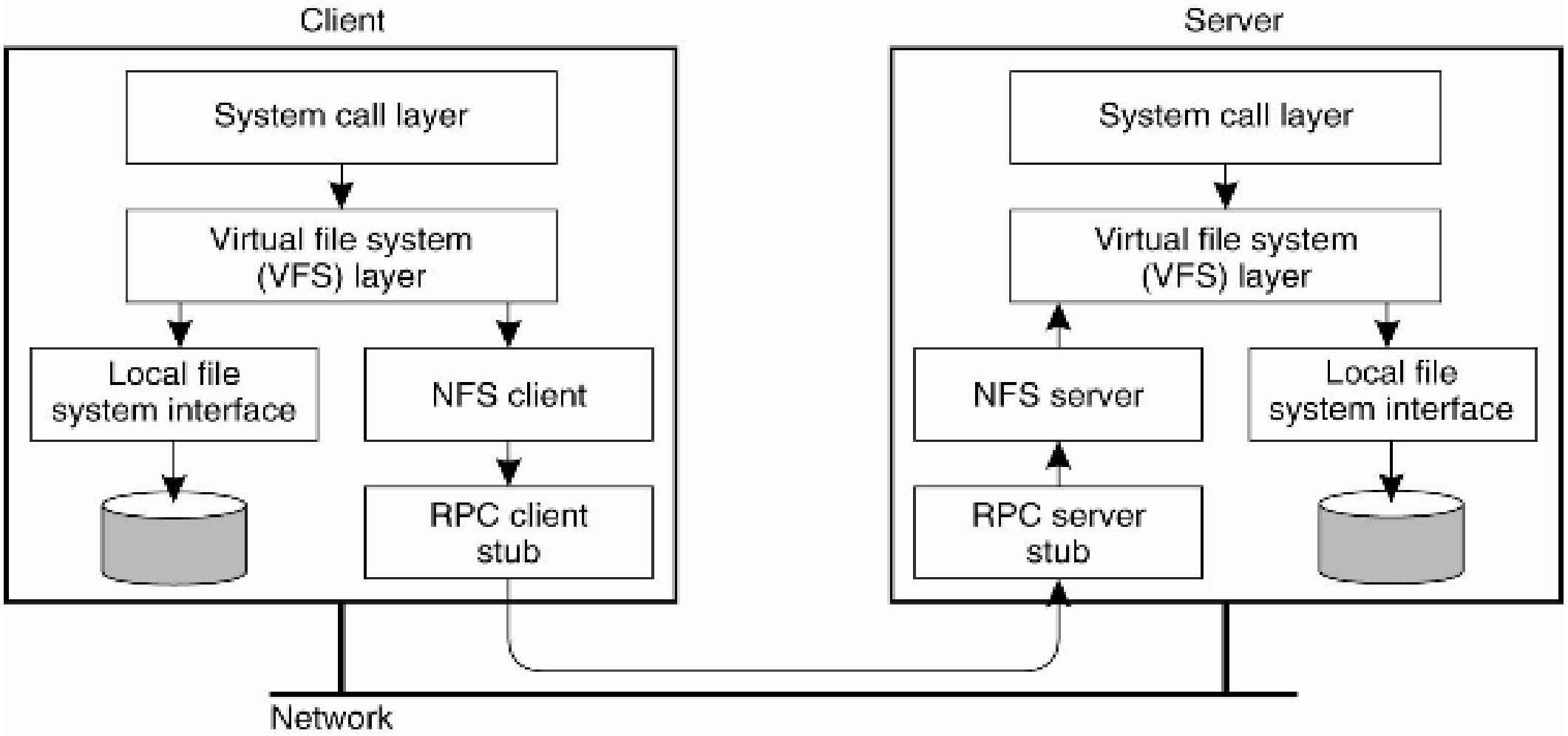
Server chứa CSDL thật sự, thực hiện việc đánh chỉ mục dữ liệu, viết các SP, tạo table...

Client là Management Studio, cho phép connect đến các CSDL khác nhau.

Đây là Thick-Server.

2) File Server (ví dụ ftp server)

Ví dụ Kiến trúc Client-Server (tt)



Ví dụ Kiến trúc Client-Server (tt)

- Kiến trúc này có 2 lớp chính là Client và Server **kết nối với nhau thông qua giao thức RPC** (Remote Procedure Call - giao thức gọi hàm từ xa).
- **VFS Layer** (lớp quản lý file ảo) thực chất **là một thư viện được tích hợp** vào, ***không phải là một tier*** vì việc trao đổi dữ liệu với lớp này được thực hiện trên bộ nhớ của cùng 1 ứng dụng chứ không phải truyền từ ứng dụng này sang ứng dụng khác.
- **Các tier là những ứng dụng chạy độc lập và trao đổi với nhau theo các giao thức riêng được định nghĩa sẵn.**

  -> Ví dụ ở đây là giao thức RPC, một số giao thức khác thường gặp như HTML, RMI, ...

II.2.2 Kiến trúc 3 lớp

- **Đặc điểm:** phân làm 3 phân hệ

  - Presentation layer: giao diện người dùng.

  - Business layer:

  + Thư viện xử lý

  + Application server

  - Data layer:

  + Dịch vụ dữ liệu

  + Database server

  Tương tác theo quy tắc "Thang máy": không tương tác vượt tầng.

- **Ưu khuyết điểm:** tương tự kiến trúc Client-Server.

II.2.2 Kiến trúc 3 lớp (tt)

- **Thư viện xử lý:** chạy chung vùng nhớ của ứng dụng sử dụng thư viện.
  - Thư viện liên kết tĩnh (.lib): đưa code vào file .exe -> kích thước file .exe lớn.
  - Thư viện liên kết động (.dll): không cần biên dịch lại nhưng phải update file .dll mới.
- **Application Server:**

  Là một dạng thư viện mới, không cần người lập trình phải update khi có thay đổi trên thư viện này.

  Không chạy chung vùng nhớ của ứng dụng, trả về các đối tượng đã được tạo ở một nơi khác -> Giao thức tạo đối tượng từ xa: RMI, COM, CORBA

Ví dụ Kiến trúc 3 lớp

Các ứng dụng web thường theo kiến trúc 3 lớp:

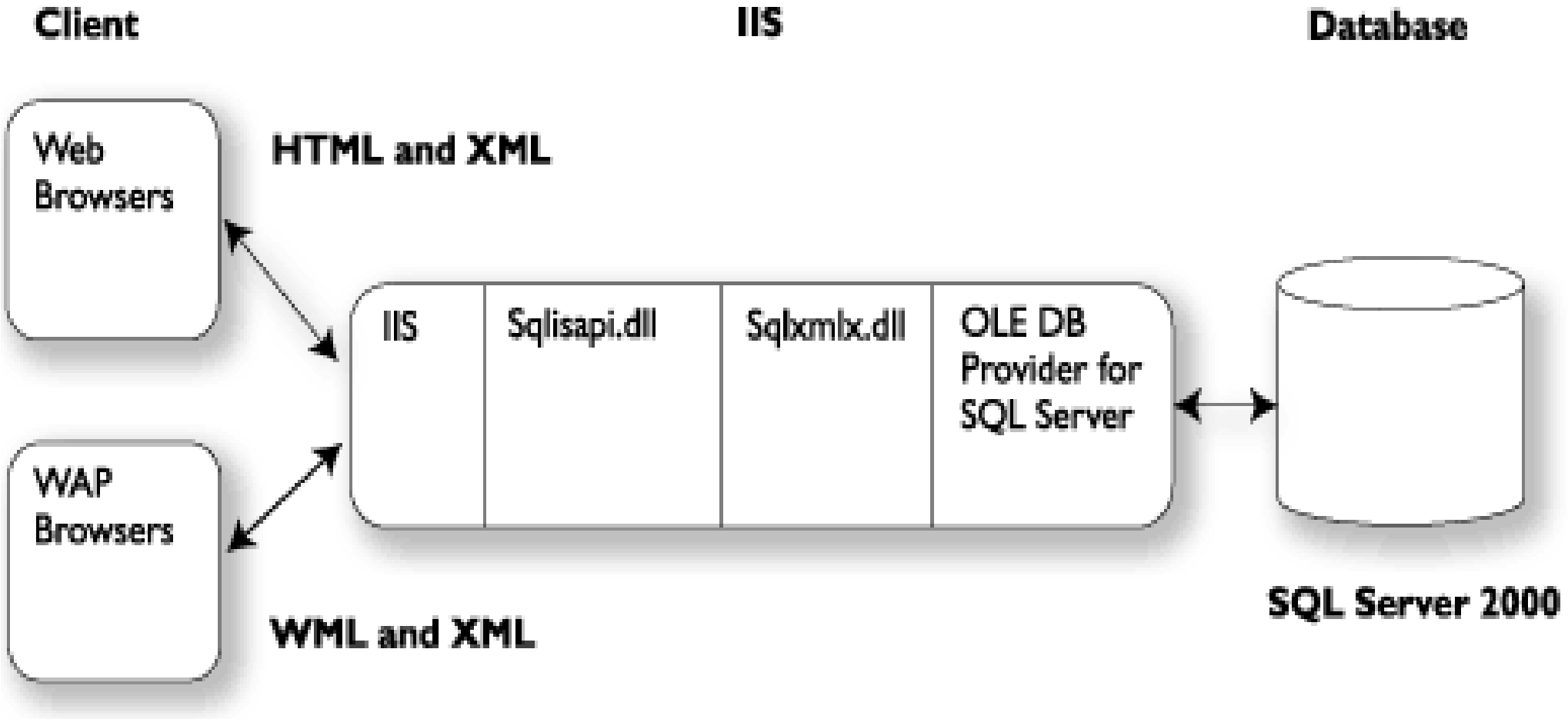
- **Presentation layer (Client):** trình duyệt.
- **Business layer (Web server):** ví dụ IIS (Internet Information Server) của Microsoft.

  + Chứa các trang .html; các file script thực hiện một số chức năng nào đó (.asp/.aspx) của ứng dụng web.

  + Đóng vai trò như một midleware.

  + Sử dụng các chức năng của Thư viện xử lý/Application server.
- **Data layer (Database server):** ví dụ SQL Server.

Ví dụ Kiến trúc 3 lớp (tt)



Ví dụ Kiến trúc 3 lớp (tt)

- **Browser** chạy với 2 **giao thức WML/HTML** kết nối với Web sever.
- **Web server** là IIS sẽ phục vụ yêu cầu của Browser thông qua giao thức WML hoặc HTML tùy theo trình duyệt.
- **Web Server** lấy dữ liệu từ SQL Server thông qua **các thư viện sqlisapi.dll và sqlxmlx.dll** theo giao thức riêng được xác lập nhờ vào trình điều khiển (driver) OLE DB Provider for SQL Server.

Triển khai ứng dụng Web

C1: Mua dịch vụ Hosting: Đăng ký một tên miền để chạy ứng dụng ở Web Browser (truy cập lên host).

C2: Demo trên máy cá nhân khi báo cáo đồ án.

C3: Nối mạng LAN:

- Một máy làm Client, chứa Browser
- Một máy làm Server, chứa IIS và SQL Server

Lưu ý: không nên xài host free vì:

- Có thể bị nhiễm virus từ server.
- Không ổn định: có thể làm hư file script...

II.2.3 Kiến trúc đa tầng

- **Đặc điểm:**

  - Mở rộng kiến trúc 3 lớp.

  - Phân làm nhiều tầng xử lý.

- **Ưu khuyết điểm:** tương tự kiến trúc Client-Server.

II.2.4 Kiến trúc Peer–to–Peer (P2P)

- **Đặc điểm:**

  - Là kiến trúc phân tán, triển khai trên nhiều máy (nút).

  - Các nút tương tác được với nhau, mỗi nút đóng vai Client – Server.

  - Chia sẻ dữ liệu và xử lý.

- **Ưu điểm:**

  - Dễ triển khai, không cần server trung tâm.

  - Không gian lưu trữ và khả năng xử lý dàn trải.

- **Khuyết điểm:**

  - Khó lập trình và quản lý dữ liệu.

II.2.4 Kiến trúc P2P (tt)

- **Kiến trúc này được phát triển thành:**
  - Grid trong mạng nội bộ;
  - Cloud trên Internet.
- **So sánh với Kiến trúc Cluster:**
  - **Kiến trúc Peer-to-Peer:** hệ thống bao gồm nhiều máy khác nhau, các máy trong hệ thống có thể khác hệ điều hành.
  - **Kiến trúc Cluster:** dùng cab kết nối main board chứa CPU, RAM, HDD, ... của các máy lại nhằm tận dụng tài nguyên của các máy (để nâng cao tốc độ xử lý...).

MVC

- MVC là thiết kế mẫu để cài đặt các xử lý của ứng dụng.
- Mỗi một xử lý sẽ có một MVC riêng cho nó, vì vậy mỗi một Form nên thực hiện một xử lý chính thôi ***(tìm kiếm/chỉnh sửa/thêm mới).***
  - **M**odel: tương tác với dữ liệu, gọi các stored procedure...
  - **V**iew: tương tác với người dùng (nhập/xuất thông tin...).
  - **C**ontroller: cài đặt các hàm xử lý.

MVC (tt)

Ưu điểm của MVC:

- Tách ra từng phần riêng biệt nên có thể thay đổi từng phần độc lập mà không sợ ảnh hưởng đến các thành phần khác (không cần biên dịch lại tất cả vì mỗi thành phần có lớp riêng tương ứng với nó): ví dụ thay đổi giao diện...
- Có thể phân công việc cho nhiều người cùng làm.

Q & A

Câu hỏi ôn tập

1) Trình bày khái niệm kiến trúc hệ thống.
2) Hãy nêu các bước thiết kế kiến trúc hệ thống.
3) Trình bày các kiến trúc hệ thống và cho ví dụ minh họa.