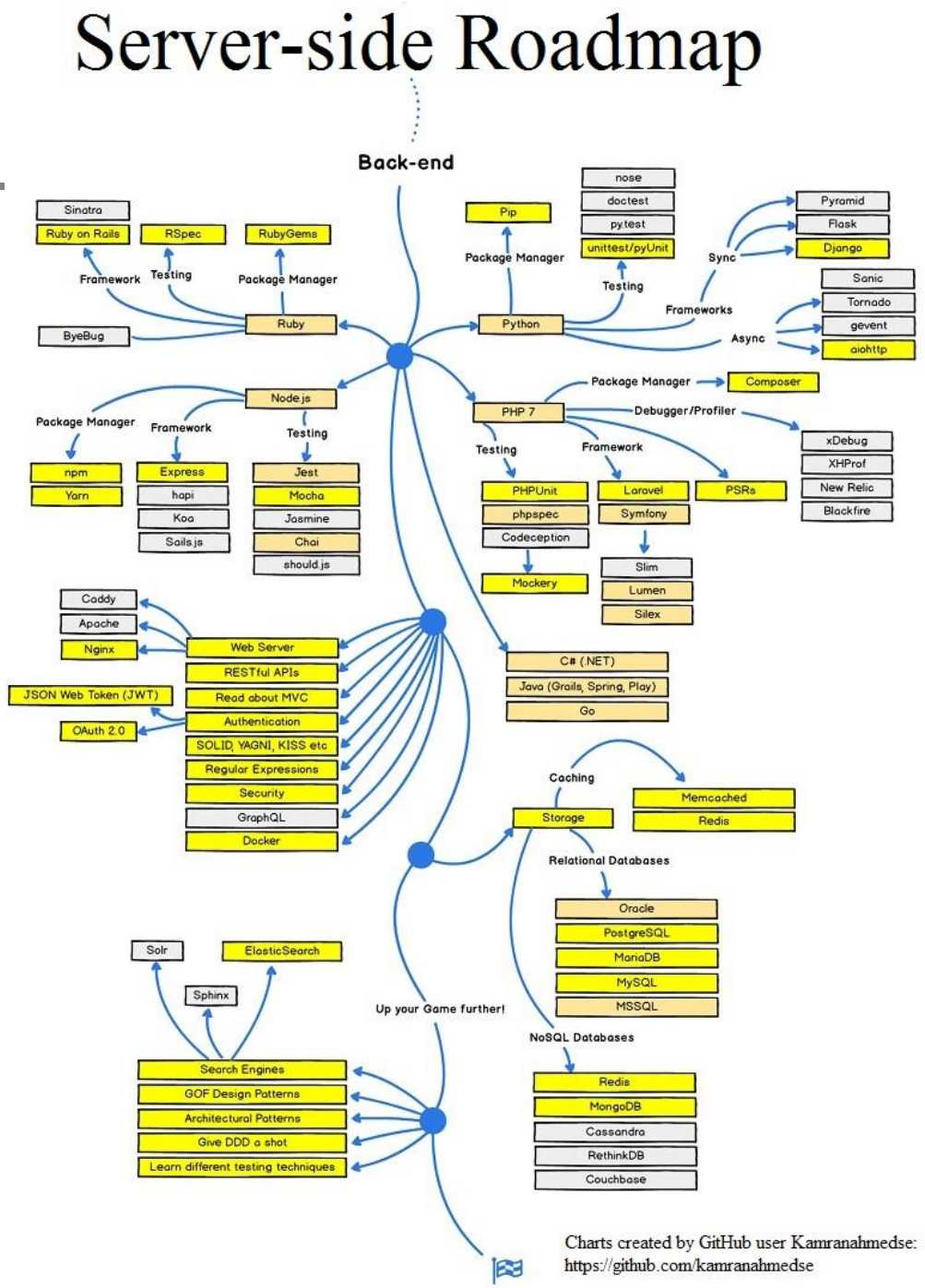
Bộ môn Công nghệ Phần mềm
Viện CNTT & TT
Trường Đại học Bách Khoa Hà Nội

CÔNG NGHỆ WEB TIÊN TIẾN

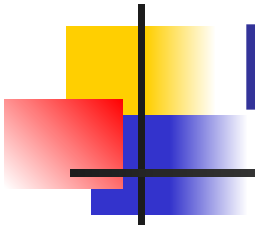
Bài 09: Tổng quan về các công nghệ Web phía server

Server-side


Server-side Roadmap
Back-end
Sinatra
Ruby on Rails
RSpec
RubyGems
Framework
Testing
Package Manager
ByeBug
Ruby
Pip
Package Manager
nose
doctest
pytest
unittest/pyUnit
Testing
Sync
Pyramid
Flask
Django
Frameworks
Async
Sanic
Tornado
gevent
aiohttp
Python
Node.js
Package Manager
Framework
Testing
npm
Yarn
Express
hapi
Koa
Sails.js
Jest
Mocha
Jasmine
Chai
should.js
Package Manager
Composer
PHP 7
Debugger/Profiler
xDebug
XHProf
New Relic
Blackfire
Testing
Framework
PHPUnit
phpspec
Codeception
Mockery
Laravel
Symfony
PSRs
Slim
Lumen
Silex
Caddy
Apache
Nginx
Web Server
RESTful APIs
Read about MVC
Authentication
SOLID, YAGNI, KISS etc
Regular Expressions
Security
GraphQL
Docker
JSON Web Token (JWT)
OAuth 2.0
C# (.NET)
Java (Grails, Spring, Play)
Go
Caching
Storage
Memcached
Redis
Relational Databases
Oracle
PostgreSQL
MariaDB
MySQL
MSSQL
NoSQL Databases
Redis
MongoDB
Cassandra
RethinkDB
Couchbase
Solr
ElasticSearch
Sphinx
Search Engines
GOF Design Patterns
Architectural Patterns
Give DDD a shot
Learn different testing techniques
Up your Game further!
Charts created by GitHub user Kamranahmedse:
https://github.com/kamranahmedse

Nội dung

1. **Các công nghệ Web phía server**
2. Kiến trúc ứng dụng Web phía server
3. Giới thiệu kiến trúc MVC

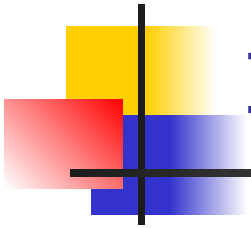


Nội dung

1.1 Server-Side Technologies

1.2 Các công việc thực hiện phía server

1.3 Sever-Side Languages

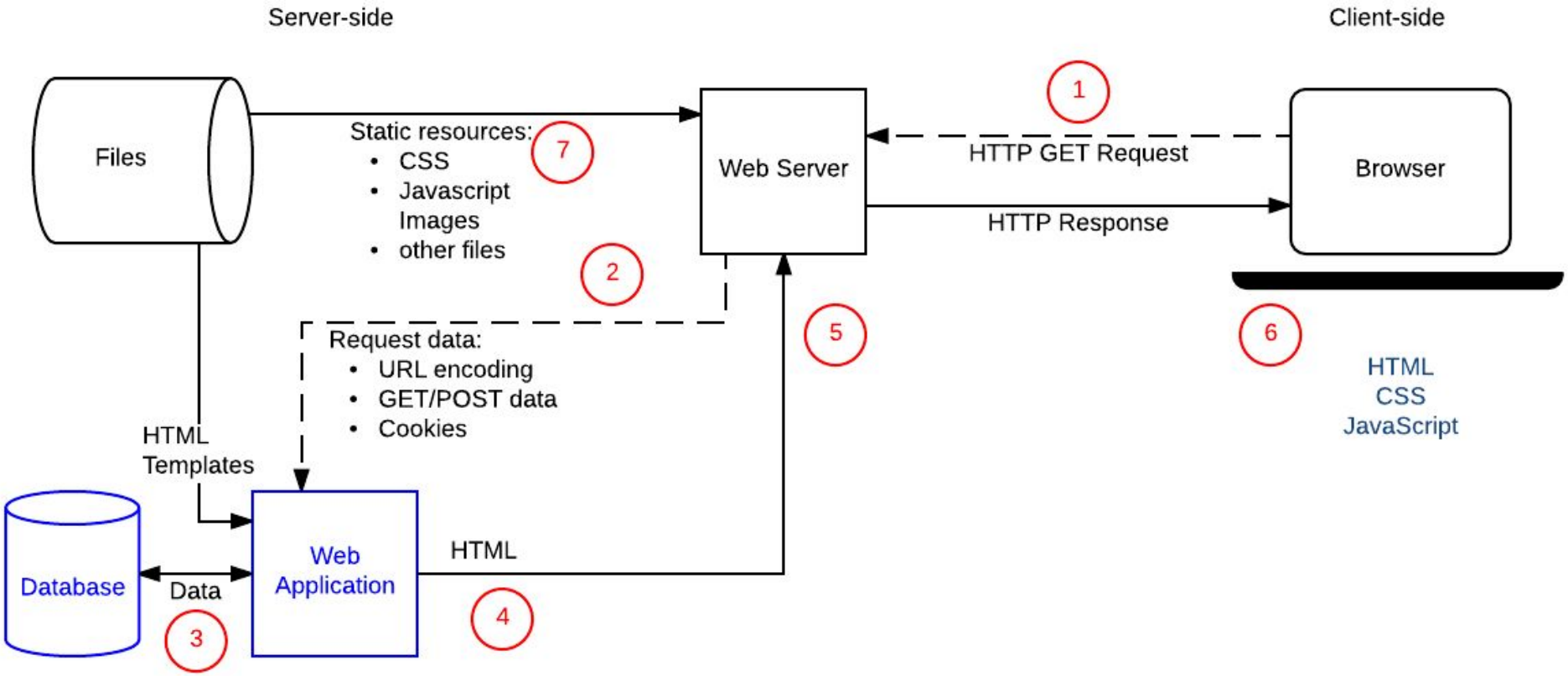


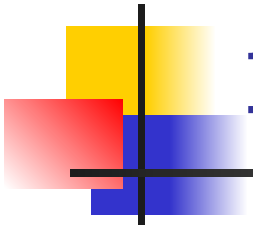
1.1 Server-Side Technologies

- Thuật ngữ công nghệ phía máy chủ có thể bao gồm:
  - Server-side languages
  - Phần mềm máy chủ web (Web server và Application Server)
  - Hệ thống quản lý cơ sở dữ liệu (Database Server)
  - Mối quan hệ client–server trong mạng máy tính
  - và nhiều công nghệ khác tùy thuộc vào ứng dụng đang được xây dựng.

1.1 Server-Side Technologies

- Những công nghệ phía máy chủ là cơ sở để xây dựng các trang web động (dynamic sites)





1.1 Server-Side Technologies

- Câu hỏi: lập trình phía máy chủ và phía máy khách có giống nhau không?
  - Có những mục đích và mối quan tâm khác nhau.
  - Chúng thường không sử dụng các ngôn ngữ lập trình giống nhau (ngoại lệ là JavaScript, có thể được sử dụng trên máy chủ và phía máy khách).
  - Chúng chạy bên trong các môi trường hệ điều hành khác nhau.

1.1 Server-Side Technologies

- So sánh client-side và server-side

Client-side	Server-side
Chạy trong trình duyệt	Hoạt động trên máy máy chủ
Cải thiện giao diện và hành vi của một trang web được hiển thị: -Lựa chọn và tạo kiểu cho các thành phần giao diện người dùng, -Tạo bố cục, -Điều hướng, -Xác thực biểu mẫu, v.v.	Chọn nội dung nào được trả về trình duyệt theo yêu cầu: -Xác thực dữ liệu và yêu cầu đã gửi, -Sử dụng cơ sở dữ liệu để lưu trữ và truy xuất dữ liệu -Gửi dữ liệu chính xác đến máy khách theo yêu cầu
HTML, CSS và JavaScript - mã này được chạy bên trong trình duyệt web và có rất ít hoặc không có quyền truy cập vào hệ điều hành cơ bản (bao gồm quyền truy cập hạn chế vào hệ thống tệp)	Có thể được viết bằng bất kỳ ngôn ngữ lập trình nào - ví dụ về các ngôn ngữ web phía máy chủ phổ biến bao gồm PHP, Python, Ruby, C # và NodeJS (JavaScript). Mã phía máy chủ có toàn quyền truy cập vào hệ điều hành máy chủ.

1.1 Server-Side Technologies

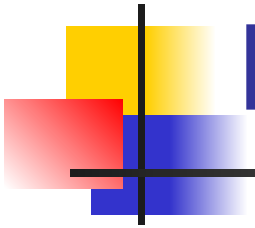
- So sánh client-side và server-side

Client-side	Server-side
Phản hồi nhanh chóng	Phản hồi chậm
Người dùng có thể xem được mã nguồn phía client	Không thể xem mã nguồn trên server, Người dùng chỉ có thể xem đầu ra được phản hồi dưới dạng HTML
Các framework web phía máy khách đơn giản hóa các nhiệm vụ trình bày và bố cục	Các framework web phía máy chủ cung cấp nhiều chức năng máy chủ web "thông thường" mà bạn có thể phải tự triển khai (ví dụ: hỗ trợ cho các phiên, hỗ trợ người dùng và xác thực, truy cập cơ sở dữ liệu dễ dàng, thư viện tạo khuôn mẫu, v.v.)

1.1 Server-Side Technologies

- So sánh client-side và server-side

Client-side	Server-side
Mã phía máy khách là tùy chọn tốt nhất cho các ứng dụng mà các phần tử trang không phụ thuộc vào cơ sở dữ liệu. Thay đổi màu sắc hoặc giao diện của các phần tử, thay đổi phông chữ hoặc ẩn các phần tử trên cơ sở lựa chọn của người dùng là những ví dụ có thể sử dụng tập lệnh phía Máy khách.	Mã hóa phía máy chủ là tùy chọn tốt hơn khi dữ liệu được trình bày cho người dùng được lưu trữ trong cơ sở dữ liệu. Xác thực người dùng, đặt chỗ trực tuyến, lưu dữ liệu do người dùng nhập là những ví dụ cần thiết lập kịch bản phía Máy chủ



Nội dung

1.1 Server-Side Technologies

1.2 Các công việc thực hiện phía server

1.3 Sever-Side Languages

1.2 Các công việc thực hiện phía server

- Các công việc thực hiện phía server:
  - Xử lý đầu vào của người dùng
  - Hiển thị các trang được yêu cầu
  - Mã hóa dữ liệu thành HTML kết hợp CSS và JS
  - Cho phép cung cấp hiệu quả thông tin phù hợp với từng người dùng và do đó tạo ra trải nghiệm người dùng tốt hơn
    - Ví dụ: Amazon sử dụng lập trình phía máy chủ để xây dựng kết quả tìm kiếm sản phẩm, đưa ra đề xuất sản phẩm được nhắm mục tiêu dựa trên sở thích của khách hàng và thói quen mua hàng trước đó, đơn giản hóa việc mua hàng, v.v.

1.2 Các công việc thực hiện phía server

- Các công việc thực hiện phía server:
  - Lưu trữ và quản lý thông tin hiệu quả
    - Lập trình phía máy chủ cho phép lưu trữ thông tin trong cơ sở dữ liệu và tự động tạo và trả về HTML và các loại tệp khác (ví dụ: PDF, hình ảnh, v.v.).
    - Cũng có thể chỉ cần trả lại dữ liệu (JSON, XML, v.v.) để hiển thị bằng các framework web phía máy khách thích hợp (điều này làm giảm gánh nặng xử lý trên máy chủ và lượng dữ liệu cần được gửi).
    - Dễ dàng được chia sẻ và cập nhật với các hệ thống kinh doanh khác (ví dụ: khi sản phẩm được bán trực tuyến hoặc tại một cửa hàng, cửa hàng có thể cập nhật cơ sở dữ liệu về hàng tồn kho của mình).

1.2 Các công việc thực hiện phía server

- Các công việc thực hiện phía server:
  - Truy cập có kiểm soát vào nội dung thông qua các cơ chế xác thực, uỷ quyền
    - Lập trình phía máy chủ cho phép các trang web hạn chế quyền truy cập đối với người dùng được ủy quyền và chỉ cung cấp thông tin mà người dùng được phép xem.
    - Ví dụ:
      - Xác thực tài khoản trước khi thực hiện đặt hàng
      - Facebook cho phép người dùng kiểm soát hoàn toàn dữ liệu của chính mình nhưng chỉ cho phép bạn bè của họ xem hoặc bình luận về nó.
      - …

1.2 Các công việc thực hiện phía server

- Các công việc thực hiện phía server:
  - Lưu trữ thông tin về phiên (session) / trạng thái
    - Lập trình phía máy chủ cho phép các nhà phát triển sử dụng session - một cơ chế cho phép máy chủ lưu trữ thông tin về người dùng hiện tại của một trang web và gửi các phản hồi khác nhau dựa trên thông tin đó
    - Ví dụ: session cho phép một trang web biết rằng người dùng đã đăng nhập trước đó và hiển thị các sản phẩm trong giỏ hàng người dung đã chọn mua hoặc lịch sử đặt hàng của họ,…

1.2 Các công việc thực hiện phía server

- Các công việc thực hiện phía server:
  - Thông báo và liên lạc (Notifications and communication)
    - Gửi thông báo chung hoặc thông báo người dùng cụ thể thông qua chính trang web hoặc qua email, SMS, hội thoại video hoặc các dịch vụ liên lạc khác.
    - Ví dụ: Amazon thường xuyên gửi e-mail sản phẩm đề xuất các sản phẩm tương tự như những sản phẩm đã mua hoặc đã xem mà người dùng có thể quan tâm,…

1.2 Các công việc thực hiện phía server

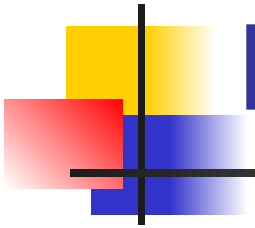
- Các công việc thực hiện phía server:
  - Phân tích dữ liệu (Data analysis)
    - Một trang web có thể thu thập nhiều dữ liệu về người dùng: họ tìm kiếm gì, mua gì, giới thiệu gì, họ ở lại mỗi trang trong bao lâu. Lập trình phía máy chủ có thể được sử dụng để tinh chỉnh phản hồi dựa trên phân tích dữ liệu này.
    - Ví dụ: Amazon và Google đều quảng cáo sản phẩm dựa trên các tìm kiếm (và mua hàng) trước đó.

1.2 Các công việc thực hiện phía server

- Lựa chọn công nghệ phù hợp phía server:
  - Miền ứng dụng (domain)
    - Mỗi công nghệ đều có mục đích sử dụng riêng và việc tận dụng lợi thế đó sẽ giúp quá trình quyết định trở nên hợp lý hơn.
    - Ví dụ:
      - .NET và PHP là lựa chọn ưu tiên cho việc xây dựng ứng dụng web thương mại và đa năng.
      - Nếu tìm kiếm phân tích dữ liệu lớn thì Python có thể là lựa chọn tốt.
      - Nếu bảo mật cấp doanh nghiệp cho ứng dụng là quan trọng thì bạn có thể sẽ muốn khám phá Java.
      - …

1.2 Các công việc thực hiện phía server

- Lựa chọn công nghệ phù hợp phía server:
  - Đặc điểm của ngôn ngữ phía server
    - Mỗi ngôn ngữ có các đặc điểm, cộng đồng, hỗ trợ và hệ sinh thái khác nhau ảnh hưởng đến quá trình ra quyết định.
  - Chuyên môn kỹ thuật
    - Việc lựa chọn công nghệ sẽ phụ thuộc vào khả năng kỹ thuật của nhóm phát triển.

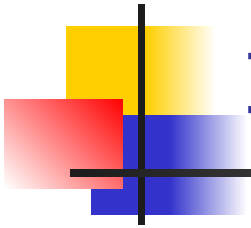


Nội dung

1.1 Server-Side Technologies

1.2 Các công việc thực hiện phía server

1.3 Sever-Side Languages



1.3 Server-Side Languages

- Đặc điểm chính:
  - Các đoạn mã được thực thi tại Web server, không phải trên Web browser
  - Các đoạn mã có thể thực thi được bởi vì có một chương trình dịch (interpreter) đã được cài đặt và kích hoạt trên Web server
  - Hầu hết là các ngôn ngữ kịch bản (scripting language)
- Phân loại:
  - Các ngôn ngữ kịch bản dựa trên tiêu chuẩn cũ: SSI (server-side includes) and CGI (common gateway interface)
  - Các ngôn ngữ kịch bản thông dụng: PHP, ASP, Perl, JSP and servlets (Java),…

1.3 Server-Side Languages

- Một số ngôn ngữ kịch bản server thông dụng:

	Perl	PHP	Python
Developer	Larry Wall etc.	Rasmus Lerdorf	Guido van Rossum, Python Software Foundation
License	GNU GPL and Artistic License	PHP-Lizenz etc.	Python Software Foundation License
Released	1987	1995	1991
Platform	independent	independent	independent
Programming paradigms	procedural, modular, also partly object-orientated	imperative, functional, object-orientated	multiparadigmatic

1.3 Server-Side Languages

- Một số ngôn ngữ kịch bản server thông dụng:

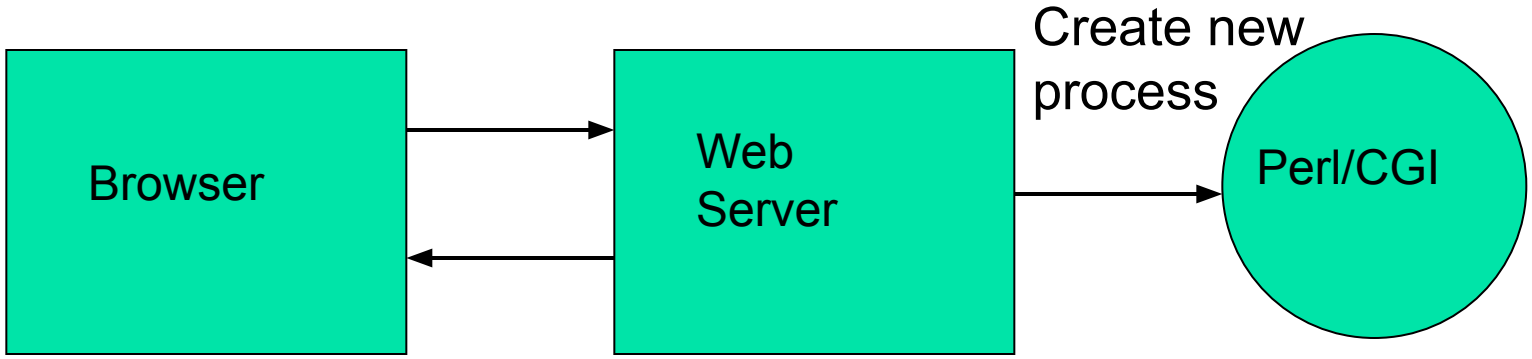
	ASP.NET	Java	Ruby
Developer	Microsoft	Sun Microsystems	Yukihiro Matsumoto etc.
License	proprietary	GNU GPL	BSD
Released	2002	1995	1995
Platform	Windows	independent	independent
Programming paradigms	object-orientated	object-orientated	multiparadigmatic

1.3.1 Common Gateway Interface (CGI)

- Giải pháp đầu tiên để làm các trang Web động
  - CGI cho phép tạo các chương trình chạy khi người dùng gửi các yêu cầu.
  - CGI script kết hợp với HTML
  - CGI script có thể được viết bằng một số ngôn ngữ từ Perl cho đến Visual Basic.
  - Hạn chế: không đảm bảo an toàn vì người khác có thể chạy chương trình trên hệ thống

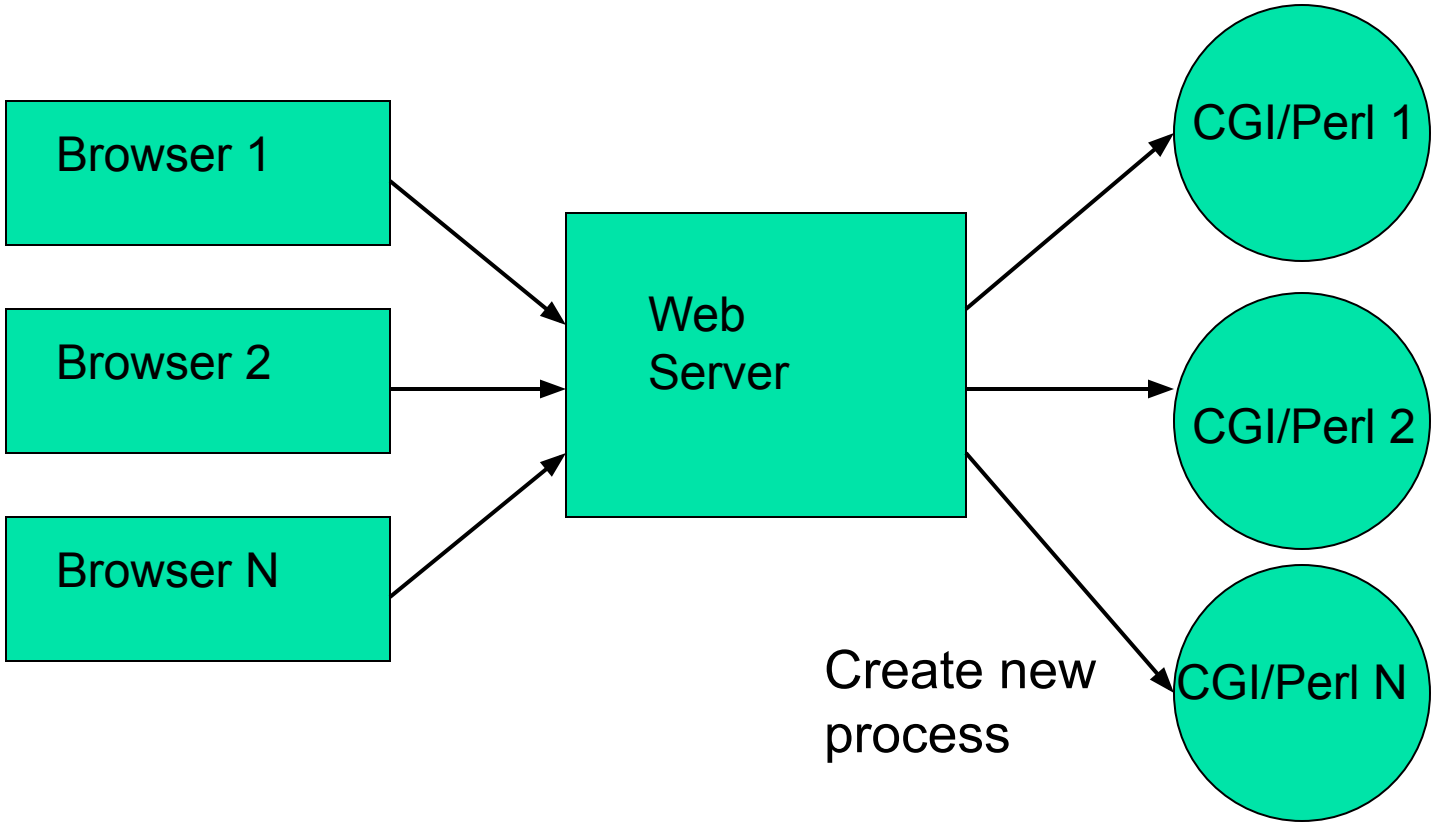
1.3.1 Common Gateway Interface (CGI)

- Kiến trúc CGI:
  - Trình duyệt gửi yêu cầu (request)
  - Máy chủ web nhận được yêu cầu
  - Đối với mỗi yêu cầu mới, máy chủ web tạo ra một tiến trình mới (new process) để thực thi chương trình CGI
  - Bất kỳ chương trình nào (tập lệnh shell, tệp loạt DOS, chương trình C, Perl) đều có thể được thực thi thông qua cơ chế này.



1.3.1 Common Gateway Interface (CGI)

- Đối với mỗi yêu cầu trình duyệt, máy chủ web phải tạo ra một tiến trình mới □ sử dụng không hiệu quả tài nguyên phía server



1.3.2 Server Side Includes (SSIs)

- Các chỉ thị bên trong một trang XHTML/HTML hướng dẫn Web server thực hiện một hành động
- Một giải pháp thay thế cho CGI
- Các lệnh SSI thường dưới dạng SGML
- Có thể dùng để:
  - Đặt các kết quả của một truy vấn cơ sở dữ liệu vào trong một trang
  - Thực thi các chương trình khác
  - Chỉ định lần cuối một tài liệu được chỉnh sửa
  - Chèn phần footer vào cuối của một trang
  - Thêm nhãn thời gian vào một trang
  - …

1.3.2 Server Side Includes (SSIs)

- Các trang được xác định bằng phần mở rộng đặc biệt (thường là ".shtml")
- Ví dụ:

```
<!--#set var="title" value="A Title" -->
<html><head><title><!--#echo var="title" --></title></head>
<body>
  <!--#include virtual="head" -->
  <table>. . . . </table>
  <!--#include virtual="foot" -->
</body>
</html>
```

1.3.3 Perl

- Ngôn ngữ thông dịch có đặc điểm là xử lý văn bản trực quan, kiểm tra kiểu lỏng lẻo, mảng liên kết, cấu trúc vòng lặp tiện dụng và xử lý tệp và môi trường đơn giản.
- Đây là ngôn ngữ kịch bản phía máy chủ phổ biến nhất trong nhiều năm
- Một tập lệnh Perl có thể được thực thi thông qua Trình thông dịch Perl từ giao diện CGI hoặc thông qua một phần mở rộng máy chủ Web nhúngPerl Interpreter trong các Máy chủ Web.

1.3.4 PHP (Hypertext Preprocessor)

- PHP có thể xem là sự giao thoa giữa Perl, C ++ và SSI.
- Hiện nay PHP là một trong những ngôn ngữ kịch bản phía máy chủ phổ biến nhất.
- Rất dễ lập trình và bảo trì, PHP có một thư viện và hệ thống API mở rộng và một số nhà cung cấp bên thứ ba (Zend, v.v.)
- Các lệnh PHP được nhúng vào trang HTML, khả năng xử lý HTTP request, response, cookie và session
- Hỗ trợ nhiều hệ quản trị CSDL, đặc biệt là MySQL

1.3.4 PHP (Hypertext Preprocessor)

- Ví dụ:

```
<?php
$title = "Sample PHP Script";
$greeting = "Welcome to Sample PHP Script";
?>
<html>
  <head>
    <title><?php echo($title) ?></title>
  </head>
  <body>
    <h1><?php echo($title) ?></h1>
    <p><?php echo($greeting) ?></p>
  </body>
</html>
```

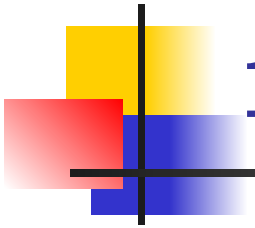
1.3.5 ASP (Active Server Pages)

- Đây là giải pháp page-centric của Microsoft.
- Nó chạy trên máy chủ IIS (Internet Information Server)
- ASP cho phép nhúng các cấu trúc động vào các trang HTML sử dụng cặp thẻ **<%** và **%>**
- Trang được lưu với phần mở rộng .asp

```
<html>
  <body>
  <%
    response.write("Hello World!")
  %>
  </body>
</html>
```

1.3.6 ASP.NET

- ASP.Net là một nền tảng dành cho phát triển web, được Microsoft phát hành và cung cấp lần đầu tiên vào năm 2002.
- Các ứng dụng ASP.Net có thể được viết bằng nhiều ngôn ngữ .Net khác nhau. Trong đó có các kiểu ngôn ngữ như C#, VB.Net,…
- Các đặc điểm của ASP.Net:
  - Code Behind Mode: tách rời thiết kế và mã code giúp duy trì ứng dụng ASP.Net trở nên dễ dàng hơn
  - State Management: ghi nhớ trạng thái của một ứng dụng tại một thời điểm
  - Caching: cải thiện hiệu suất làm việc cho ứng dụng



1.3.7 JSP/Servlets

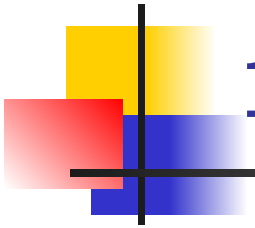
- Servlet: các đoạn mã Java được chạy trên một ứng dụng phía Server
  - Servlet là một bước tiến lớn, nó đưa ra một thư viện hàm API trên Java và một thư viện hoàn chỉnh để thao tác trên giao thức HTTP
- JavaServer Page (JSP) được thiết kế để phân tách qua trình xử lí khỏi quá trình biểu diễn, tập trung vào cách bố trí (layout)
- Chúng được biên dịch thành các servlet và được triển khai trong một Web Container riêng biệt.

1.3.7 JSP/Servlets

- Ví dụ một Servlet xử lý các request dạng GET và một trang JSP

```
public void doGet (HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {
   String title = "Hello World Servlet";
   res.setContentType("text/html");
   ServletOutputStream out = res.getOutputStream();
   out.println("<html>");
   out.println("<head><title>+title+</title></head>");
   out.println("<body>");
   out.println("<h1>+title+</h1>");
   out.println("</body></html>");
 }
```

```
<HTML>
  <HEAD>
    <% String title = "Hello World JSP"; %>
    <TITLE><%= title %></TITLE>
  </HEAD>
  <BODY>
    <H1><%= title %></H1>
  </BODY>
</HTML>
```



1.3.8 Python

- Python là ngôn ngữ lập trình xu hướng mới nhất để phát triển back end.
- Python là một trong những ngôn ngữ kịch bản phổ biến nhất, nổi tiếng vì nó dễ học và dễ sử dụng cho cả người mới bắt đầu cũng như các lập trình viên có kinh nghiệm và là mã nguồn mở.
- Nó nhấn mạnh vào khả năng đọc và có cách tiếp cận hướng đối tượng chủ yếu được sử dụng để phân tích thống kê.
- Python có thể sử dụng để viết mã cho các dự án nhỏ cũng như lớn.

1.3.9 Ruby

- Ruby là một ngôn ngữ lập trình được định kiểu động. Ruby có các đặc trưng của một ngôn ngữ hướng đối tượng. Nó đóng vai trò là nền tảng cho các ngôn ngữ miền cụ thể.
- Ruby có thể được thực hiện trên tất cả các nền tảng.
- Ruby on Rails ("RoR") là web-application framework được triển khai bằng Ruby.

```
# Ruby hiểu ý của bạn
# Bạn có thể làm toán
# trên toàn bộ chuỗi Array
thành_phố = [ "Sài Gòn", "Huế", "Đà Nẵng", "Hà Nội" ]
đã_thăm = ["Huế", "Hà Nội"]
put "Tôi phải " + "ghé thăm " + "các thành phố:", thành_phố - đã_thăm
```

1.3.10 Javascript on server

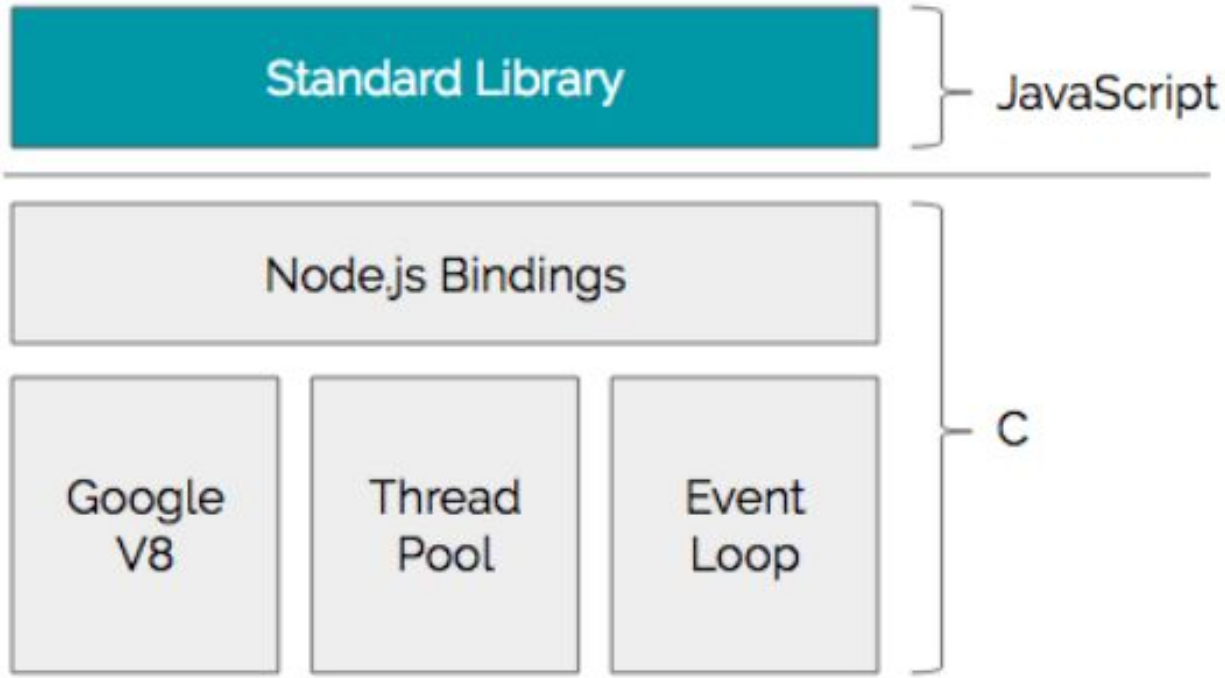
- Đây là ngôn ngữ lập trình được coi là nền tảng của sự phát triển web.
- Javascript được sử dụng cho cả ứng dụng front end và back end.
- Kể từ giữa những năm 2000, ngày càng có nhiều sự quan tâm đến JavaScript phía máy chủ.
  - Một nguyên nhân khác là thực tế là nhiều nhà phát triển web đã quen thuộc với JavaScript phía máy khách, như một phần của việc viết giao diện người dùng của các ứng dụng web. Chuyển sang JavaScript phía máy chủ có thể cho phép một tổ chức tận dụng tốt hơn nguồn nhân lực có sẵn.

1.3.10 Javascript on server

- Node.js: là một môi trường thời gian thực thi JavaScript (Framework) mã nguồn mở, đa nền tảng, thực thi mã JavaScript bên ngoài trình duyệt web.
- Node.js cho phép các nhà phát triển sử dụng JavaScript để viết các công cụ dòng lệnh và để tạo kịch bản phía máy chủ — chạy các tập lệnh phía máy chủ để tạo ra nội dung trang web động trước khi trang được gửi đến người dùng trình duyệt web.
- Node.js đại diện cho mô hình JavaScript ở khắp mọi nơi, thống nhất phát triển ứng dụng web xung quanh một ngôn ngữ lập trình duy nhất, thay vì các ngôn ngữ khác nhau cho các tập lệnh phía máy chủ và phía máy khách.

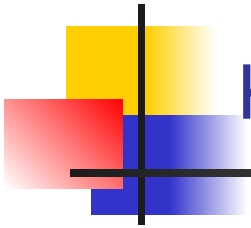
1.3.10 Javascript on server

- Kiến trúc Node.js
  - Thư viện chuẩn của nó được viết bằng JavaScript. Các ràng buộc với hệ điều hành cơ bản nằm trong C. Tất cả JavaScript đều chạy trên engine V8 của Google.



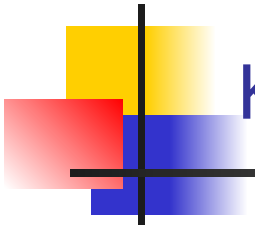
Nội dung

1. Các công nghệ Web phía server
2. **Kiến trúc ứng dụng Web phía server**
3. Giới thiệu kiến trúc MVC



Kiến trúc ứng dụng Web phía server

- 1 tầng (Single tier)
- 2 tầng (Two tier)
- 3 tầng (Three tier)
  - Dựa trên RPC
  - Dựa trên Remote object
- 3 tầng (HTML browser và Web server)
- Máy chủ ứng dụng độc quyền (Proprietary application server)
- Máy chủ ứng dụng chuẩn (Standard application server)

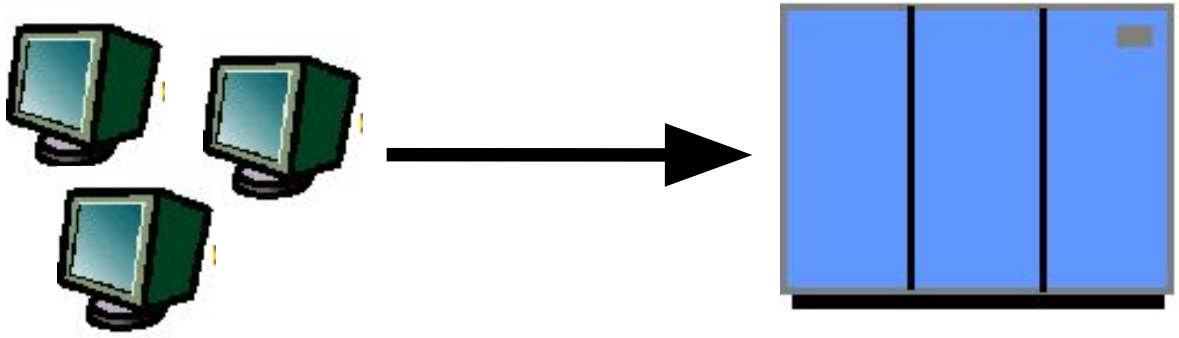


Kiến trúc logic của (Enterprise Applications)

- Các thành phần của một enterprise application
  - Presentation logic
  - Business logic
  - Data access logic (và data model)
  - System services

Single Tier (Mainframe-based)

- Dumb terminals kết nối trực tiếp với mainframe
- Theo mô hình tập trung (ngược với mô hình phân tán)
- Các xử lý Presentation, business, và data access được thực hiện duy nhất trong 1 ứng dụng ở mainframe

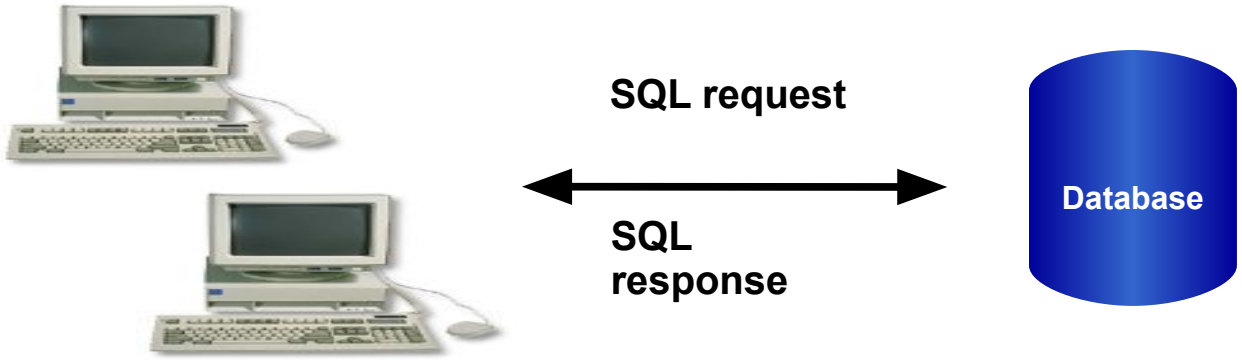


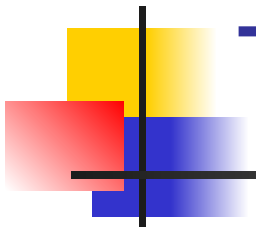
Single-Tier: Ưu nhược điểm

- Ưu điểm:
  - Không cần quản lý phía client
  - Nhất quán dữ liệu (Data consistency)
- Nhược điểm:
  - Các thao tác (presentation, data model, business logic) được mix vào lẫn nhau □ khó bảo trì & tái sử dụng code.

Two-Tier

- Fat clients giao tiếp với database phía sau
  - Gửi truy vấn SQL, nhận về dữ liệu thô
- Presentation, Business logic và Data Model processing logic đều nằm trong ứng dụng client



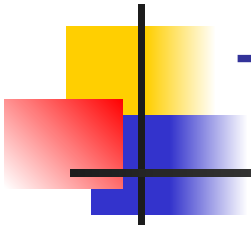


Two-Tier

- Thuận lợi:
  - DB product independence
- Hạn chế:
  - Presentation, data model, business logic được mix lẫn nhau ở client ☐ khó cập nhật & bảo trì
  - Data Model đính chặt với mọi client: nếu CSDL thay đổi ☐ tất cả clients đều hỏng
  - Các cập nhật phải được tiến hành trên tất cả các clients ☐ ác mộng trong bảo trì hệ thống
  - Tạo DB connection cho tất cả client ☐ Tốn tài nguyên, khó mở rộng
  - Trao đổi dữ liệu thô (Raw data) ☐ tốn băng thông

Three-Tier (RPC based)

- Thinner client: business & data model được tách biệt với presentation
  - Business logic và data access logic nằm trên server trung gian (middle tier server), client xử lý presentation
- Middle tier server cần xử lý các dịch vụ hệ thống
  - Điều khiển tương tranh, đa luồng, transaction, security, persistence, multiplexing, performance, ...



Three-tier (RPC based): ưu nhược điểm

- Ưu điểm:
  - Business logic thay đổi linh động hơn so với mô hình 2-tier
    - (Nằm trên middle-tier server)
- Nhược điểm:
  - Độ phức tạp cao ở middle-tier server
  - Client và middle-tier server liên kết khá chặt (So với mô hình three-tier object based)
  - Chưa thực sự tái sử dụng được Code (So với mô hình three-tier object based)

Three-Tier (Remote Object based)

- Business logic và data model phát triển theo mô hình ĐT
  - Business logic và data model được mô tả "trừu tượng" (interface language)
- Mô hình đối tượng được sử dụng: CORBA, RMI, DCOM
  - Interface language trong CORBA là IDL
  - Interface language trong RMI là Java interface

Three-tier (Remote Object based): ưu nhược điểm

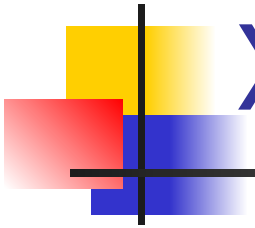
- Ưu điểm:
  - Kết nối linh động hơn mô hình RPC
  - Tái sử dụng Code tốt hơn
- Nhược điểm:
  - Vẫn có độ phức tạp ở server trung gian

Three-Tier (Web Server)

- Browser xử lý presentation logic
- Browser giao tiếp với Web server qua giao thức HTTP
- Business logic và data model được xử lý theo công nghệ "Sinh nội dung động" (CGI, Servlet/JSP, ASP)

Three-tier (Web Server based): ưu nhược điểm

- Ưu điểm:
  - Đa dạng cho thiết bị client
    - Điện thoại di động hỗ trợ J2ME, …
  - Không cần quản lý client
- Nhược điểm:
  - Vẫn còn phức tạp ở tầng trung gian



Xu hướng

- Chuyển kiến trúc single-tier hoặc two-tier thành kiến trúc multi-tier
- Chuyển từ mô hình nguyên khối (monolithic model) sang mô hình ứng dụng hướng đối tượng
- Chuyển ứng dụng client thành HTML-based client

Các vấn đề và gải pháp nổi bật

- Phức tạp ở middle tier server
- Cần nhân bản các dịch vụ hệ thống, giải quyết các vấn đề về
  - Concurrency control, Transactions
  - Load-balancing, Security
  - Resource management, Connection pooling
- Giải pháp?
  - Cần có 1 container chung xử lý toàn bộ dịch vụ hệ thống ở trên
  - Giải pháp độc quyền và giải pháp mở, chuẩn

Giải pháp độc quyền-Proprietary Solution

- Sử dụng mô hình "component và container"
  - Components đảm nhiệm các business logic
  - Container cung cấp môi trường thực thi kèm các dịch vụ hệ thống
- Components và container có giao kết chặt chẽ, rõ ràng nhưng theo cách thức của bên cung cấp
  - ☐ Vấn đề: phụ thuộc vào nhà cung cấp
- Ví dụ: Tuxedo, .NET

Giải pháp mở & chuẩn - Open and Standard Solution

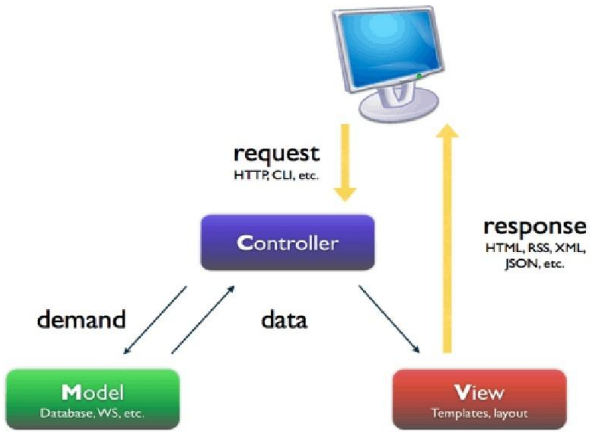
- Sử dụng mô hình "component và container" trong đó container cung cấp các dịch vụ hệ thống theo chuẩn.
- J2EE: là 1 chuẩn như thế, cho phép linh động trong code
  - Dựa trên công nghệ Java và standard-based Java programming APIs

Nội dung

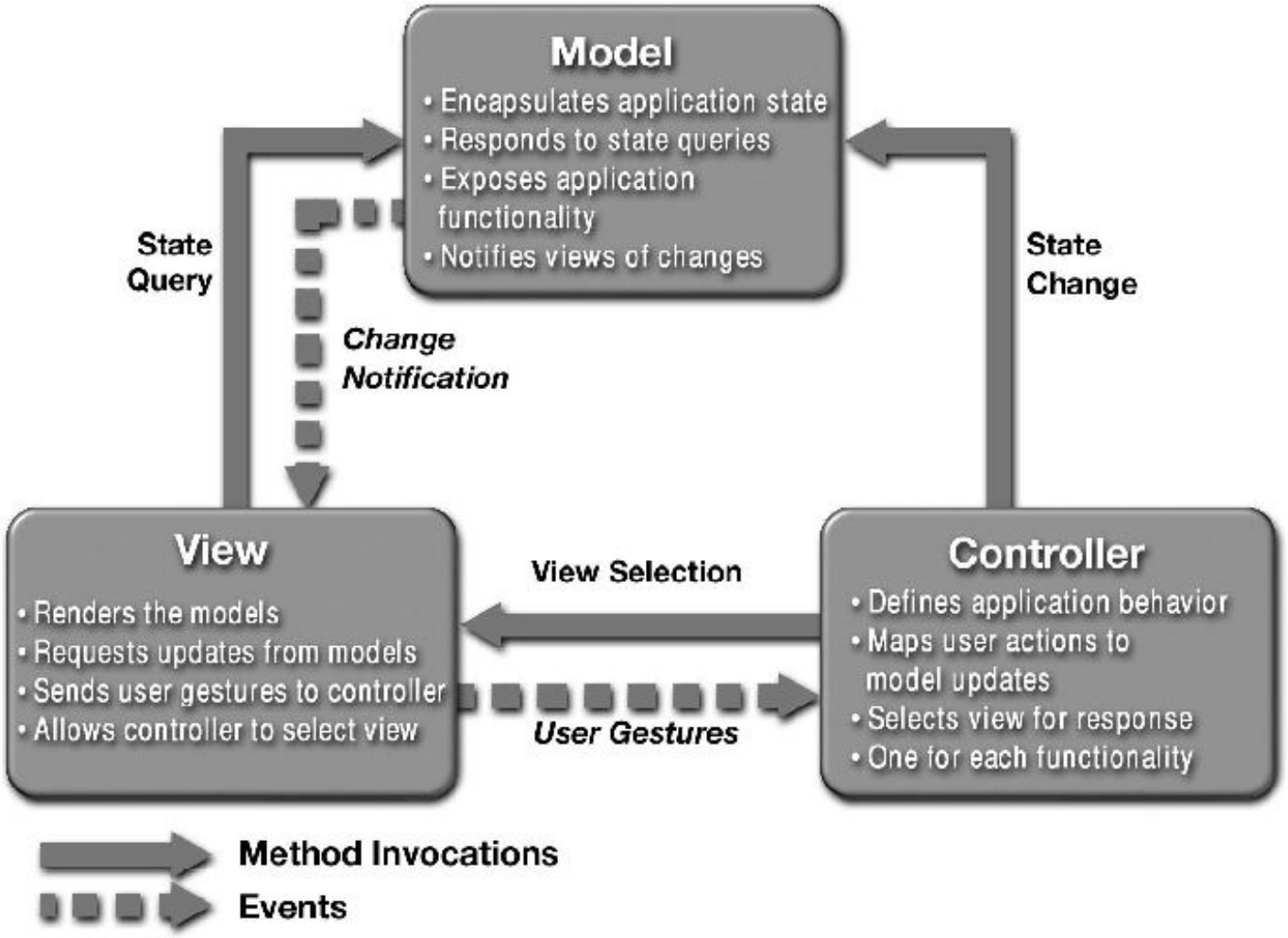
1. Các công nghệ Web phía server
2. Kiến trúc ứng dụng Web phía server
3. **Giới thiệu kiến trúc MVC**

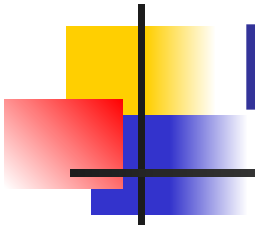
MVC Pattern

- MVC (viết tắt của Model-View-Controller) là một mẫu kiến trúc phần mềm, chia một ứng dụng thành ba phần tương tác được với nhau để tách biệt giữa cách thức mà thông tin được xử lý nội hàm và phần thông tin được trình bày và tiếp nhận từ phía người dung.
- MVC giúp cho người phát triển phần mềm cô lập các nguyên tắc nghiệp vụ và giao diện người dùng một cách rõ ràng hơn, tạo nhiều thuận lợi cho việc phát triển và bảo trì



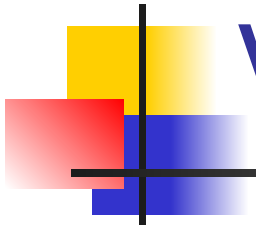
MVC Pattern


Model
• Encapsulates application state
• Responds to state queries
• Exposes application functionality
• Notifies views of changes
State Query
Change Notification
State Change
View
• Renders the models
• Requests updates from models
• Sends user gestures to controller
• Allows controller to select view
View Selection
User Gestures
Controller
• Defines application behavior
• Maps user actions to model updates
• Selects view for response
• One for each functionality
Method Invocations
Events



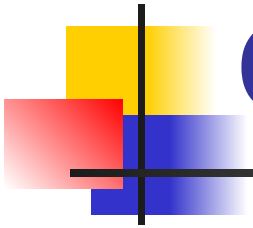
Model

- Model (Business process layer)
  - Mô hình hóa dữ liệu và hành vi (data & behavior) trong xử lý nghiệp vụ (business process)
  - Chịu trách nhiệm
    - Thực hiện các truy vấn DB
    - Tính toán trong các nghiệp vụ
    - VD: Xử lý các orders
  - Đóng gói dữ liệu và hành vi, độc lập với tầng presentation



View

- View (Presentation layer)
  - Hiển thị thông tin tùy thuộc vào loại client
  - Biểu diễn kết quả của tầng business logic (Model)
  - Không cần quan tâm làm thế nào có được thông tin, hoặc thông tin ở đâu (Model chịu trách nhiệm)



Controller

- Controller (Control layer)
  - Kết nối tương tác của người dùng với các nghiệp vụ cung cấp phía sau
  - Chọn ra cách biểu diễn phù hợp
    - Ví dụ: ngôn ngữ, biến đổi định dạng thông tin theo vùng, quyền hạn người dùng
  - Một request tới ứng dụng sẽ chuyển cho tầng Control.
    - Tầng này quyết định request được xử lý như thế nào, và thông tin sẽ được trả lại như thế nào

MVC Pattern

- Ưu điểm:
  - Nhiều chế độ View có thể được thực hiện cho các Model
  - Phân vùng nhiệm vụ giúp Lập trình viên chuyên sâu trong việc phát triển và nâng cấp trong tương lai.
  - Lý thuyết MVC hoạt động có hành vi ghép thấp giữa các mô hình, khung nhìn và bộ điều khiển.
  - Nhiều Lập trình viên có thể cùng làm việc trên Model, View, Controller cùng một lúc. Điều này giúp việc gia tăng nhân lực để tăng tốc độ dự án là khả thi.
  - Các View cho một mô hình cần thiết được nhóm lại với nhau