Web Development Concepts

Paul Fodor

CSE316: Fundamentals of Software Development

Stony Brook University

http://www.cs.stonybrook.edu/~cse316

Topics

- Short History of the Web
- Web Technologies
- Structure of a Web Application
- MVC Paradigm
- Web Application Architectures
- Web Services
  - REST API
  - SOAP API

Web In the Beginning…

- European Organization for Nuclear Research (CERN) in Switzerland:
  - 1989 – Tim Berners-Lee working at CERN proposed "a large hypertext database with typed links"
  - Generated little interest
  - Encouraged by boss to develop it anyway
- By late 1990
  - HTTP – HyperText Transfer Protocol (0.9)
  - HTML – HyperText Markup Language
  - WorldWideWeb (first web browser)
  - CERN httpd (First web server)

Early days

- 1992-1995:
  - Web was used in the beginning largely by physicists and other scientists/academia
  - Univ of Kansas adapted a text-based browser (Lynx)
  - Web first widely popularized by NCSA Mosaic released in 1993
    - First widely available graphical browser
  - Netscape Navigator developed by Mosaic's original writer (Marc Andreesen) and form SGI CEO James H. Clark

Commercialization of the Web

- 1996-1998:
  - Many companies began to offer goods and services over the web
  - Alternative to 'Brick and Mortar' business model
- 1999-2001:
  - 'Dot-com' bubble
  - 'Dot-com' bust

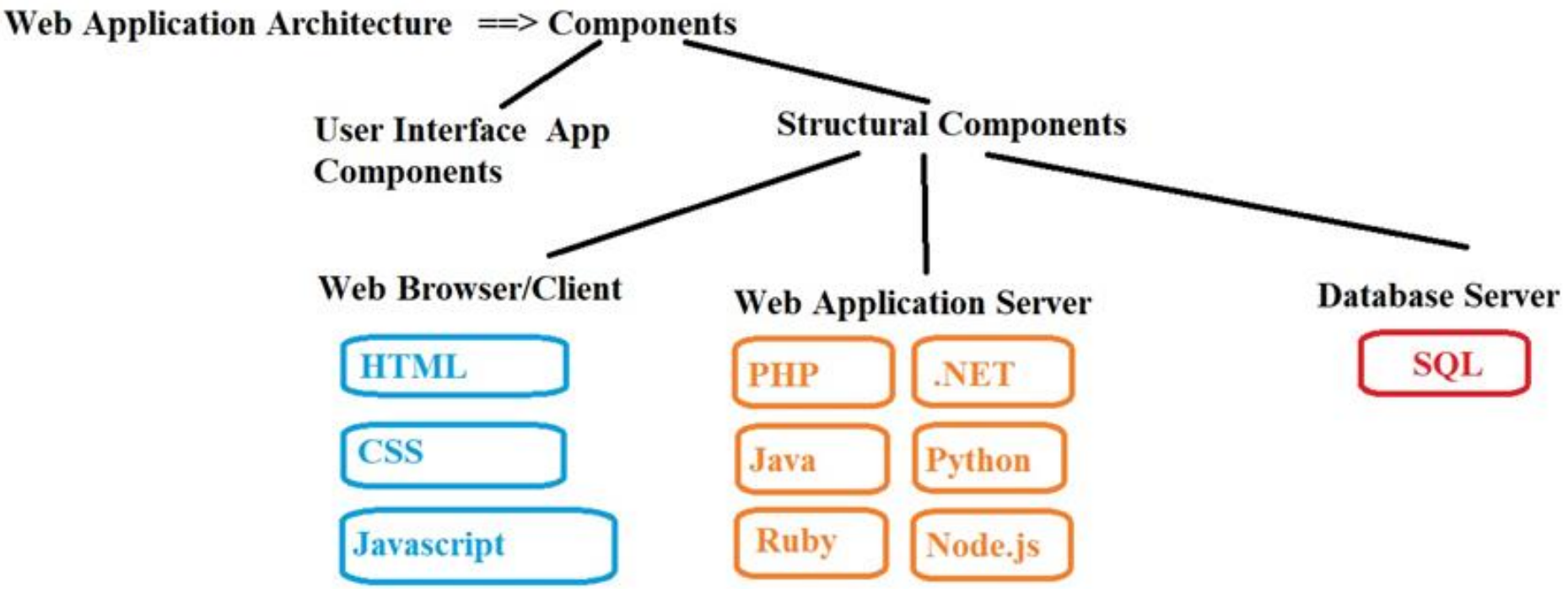
Ubiquity

- 2002-present:
  - Some companies survived
    - Airline booking sites
    - eBay auction site
    - Amazon
  - Search engines began coming on line
    - Google – profitable due to keyword-based advertising
  - Social Networking sites
    - MySpace
    - Facebook

Web Technologies

- Web 2.0
  - Changed web from a read-only media to a read/write media
  - New ideas for sharing content
    - Weblogs (blogs)
    - Social Media
    - RSS – Really Simple Syndication
    - Wiki
  - Web technologies added to help enhance content
    - XML/XHTML/HTML 5/AJAX
    - CSS – Cascading Style Sheets
    - Php

Structure of a Web Application



Web Application Architectures

- What is a ***Web Application Architecture***?
  - Describes interactions between apps, databases, and middleware
  - Assures multiple applications can work together
  - Includes aspects to assist with efficiency, reliability, scalability, security, and robustness
- Examples
  - PHP Web Application Architecture
  - Java Web Application Architecture
  - .NET Web Application Architecture
  - Python Web Application Architectures
  - Node.js Web Application Architecture

PHP Web Application Architecture

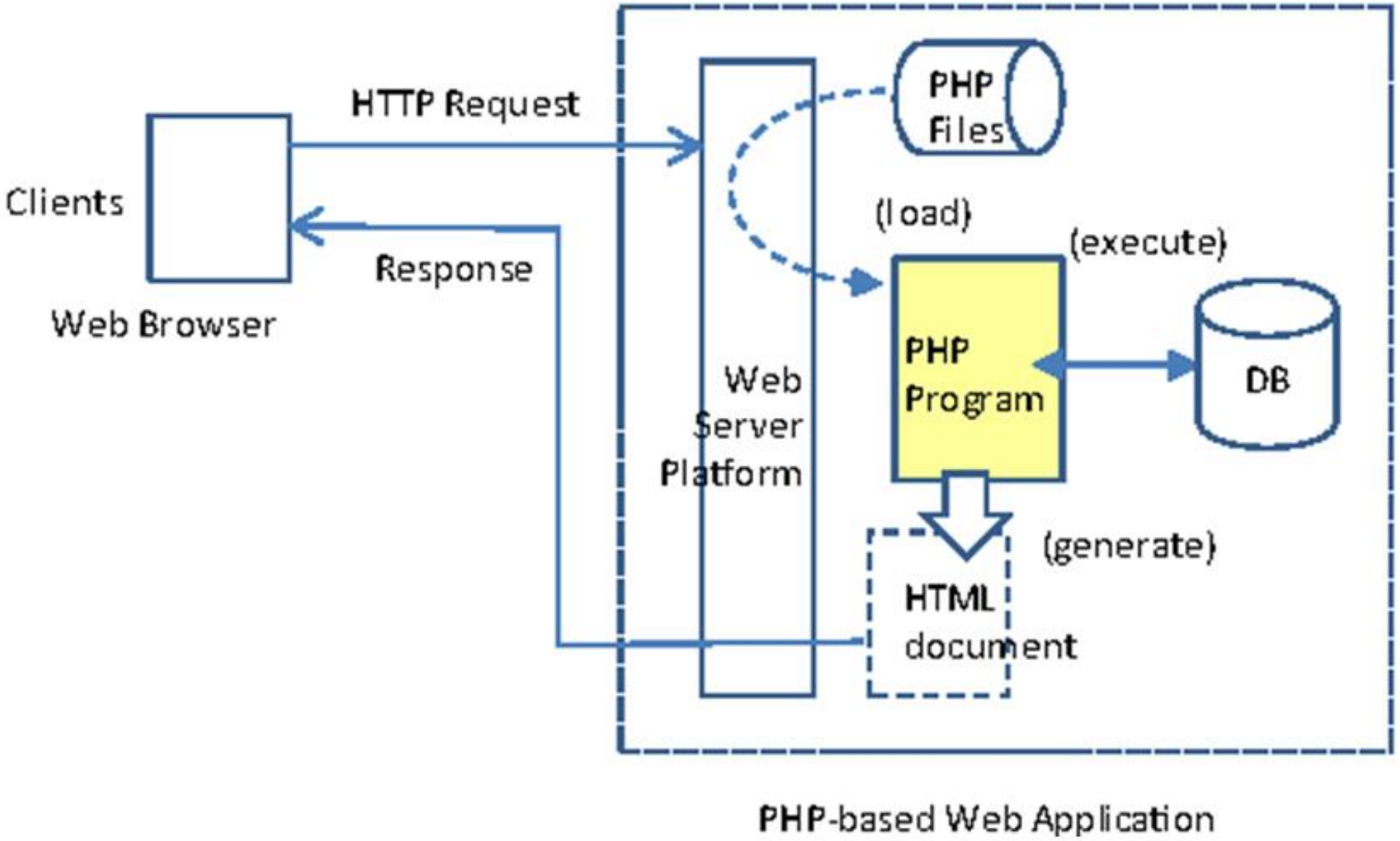
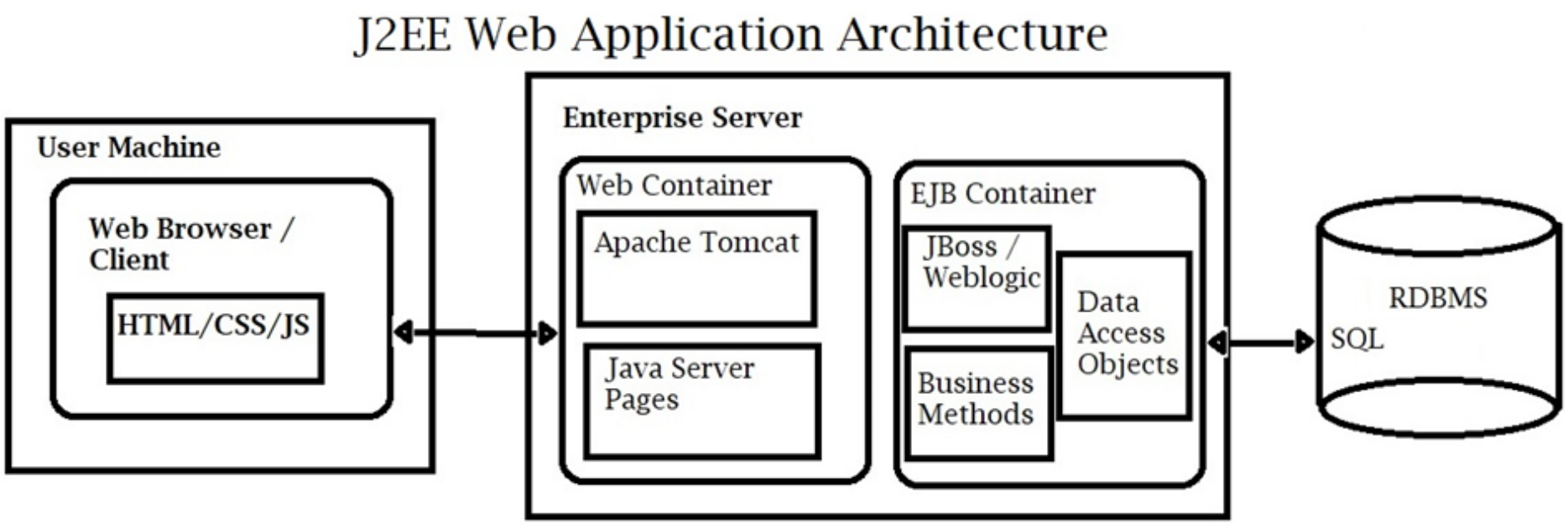


Figure from: https://www.semanticscholar.org/paper/An-Architecture-of-Dynamically-Adaptive-PHP-based-Nakajima/f6cfb2dcaf82cc228af32132ef840dbe2779e2c6/figure/0

Java Web Application Architecture



.NET Web Application Architecture

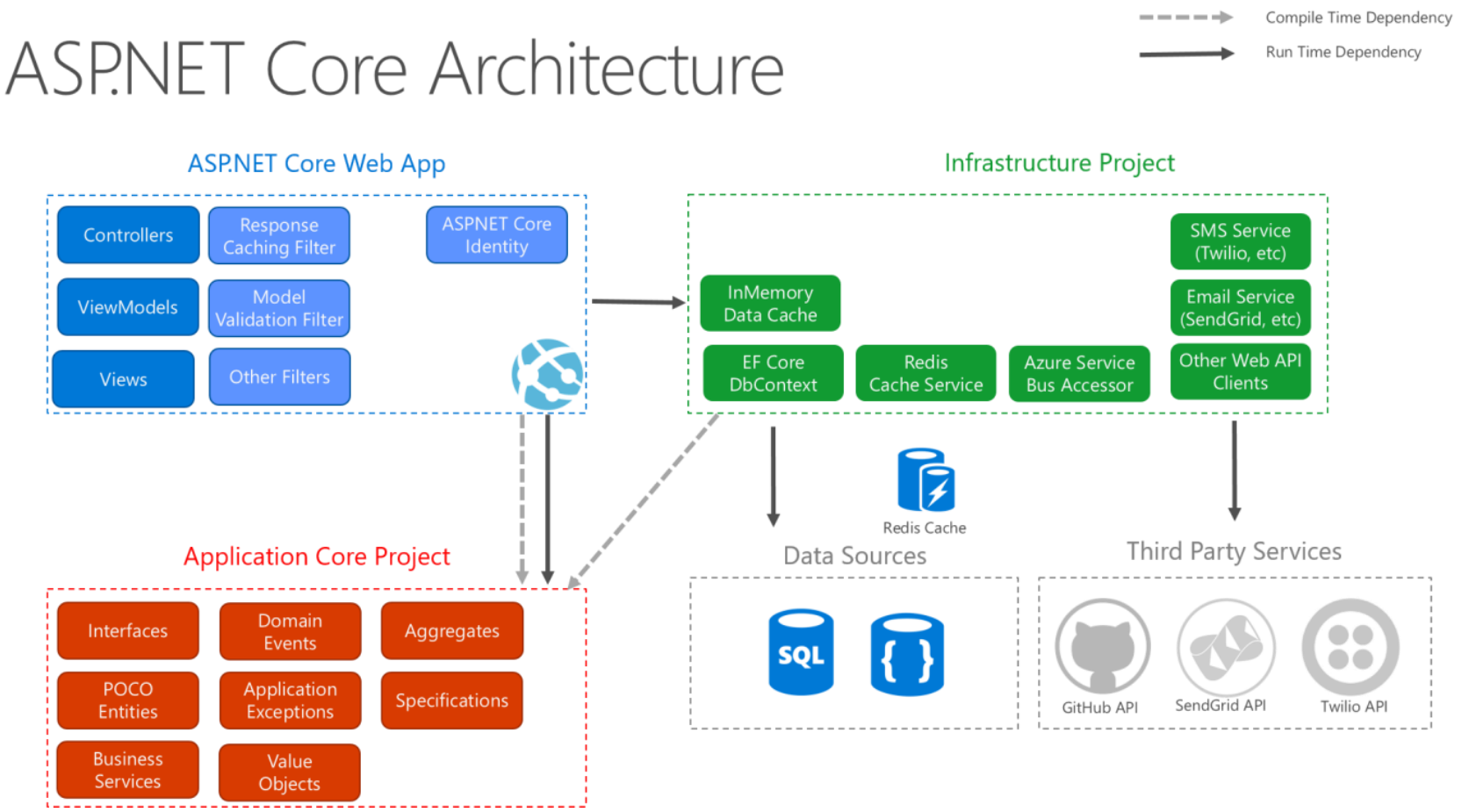
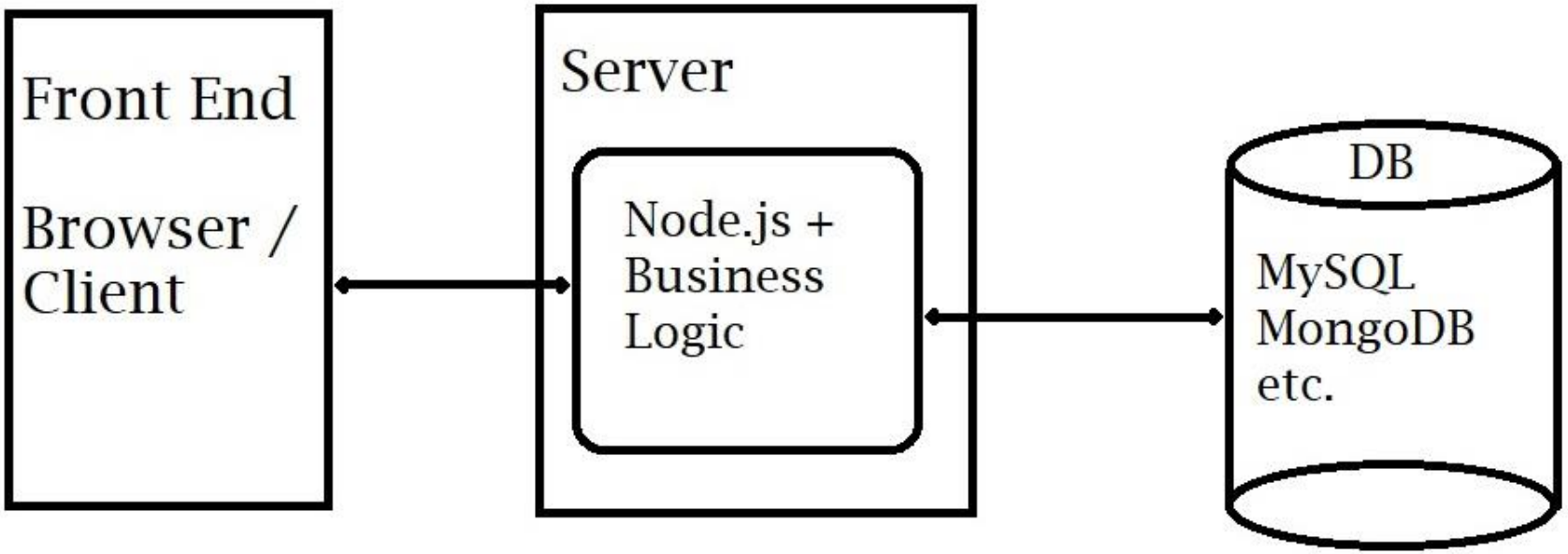


Diagram From: https://docs.microsoft.com/en-us/dotnet/standard/modern-web-apps-azure-architecture/common-web-application-architectures

Node.js Web Application Architecture



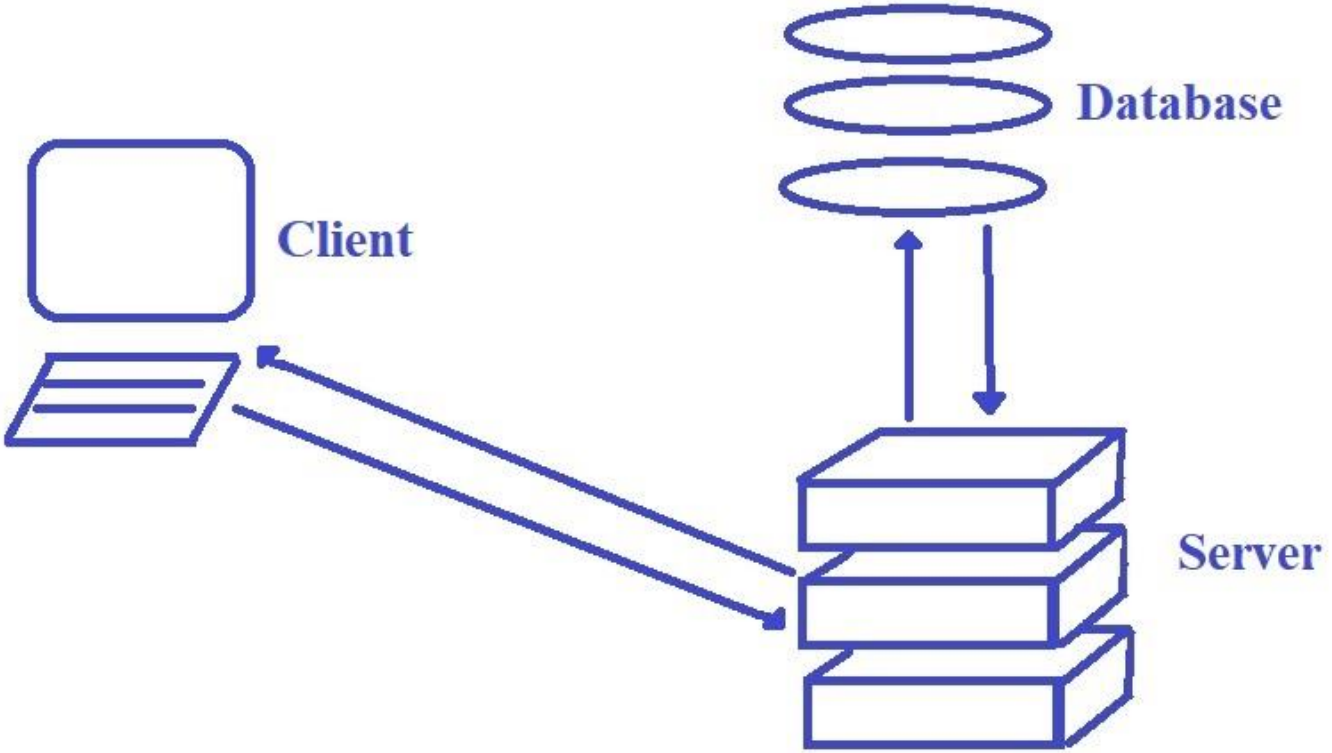
Web Development Frameworks

- A web framework (WF) or web application framework (WAF) is a software framework that is designed to support the development of web applications including web services, web resources, and web APIs.
  - Early hypertext consisted of hand-coded HTML
  - In 1993, the Common Gateway Interface (CGI) standard was introduced for interfacing external applications with web servers, to provide a dynamic web page that reflected user inputs
    - Original implementations of the CGI interface typically had adverse effects on the server load however, because each request started a separate process
  - In 1995, fully integrated server/language development environments first emerged and new web-specific languages were introduced: PHP and Active Server Pages
  - "Full stack" frameworks began to appear: CakePHP, Laravel [PHP], Rails [Ruby], etc.

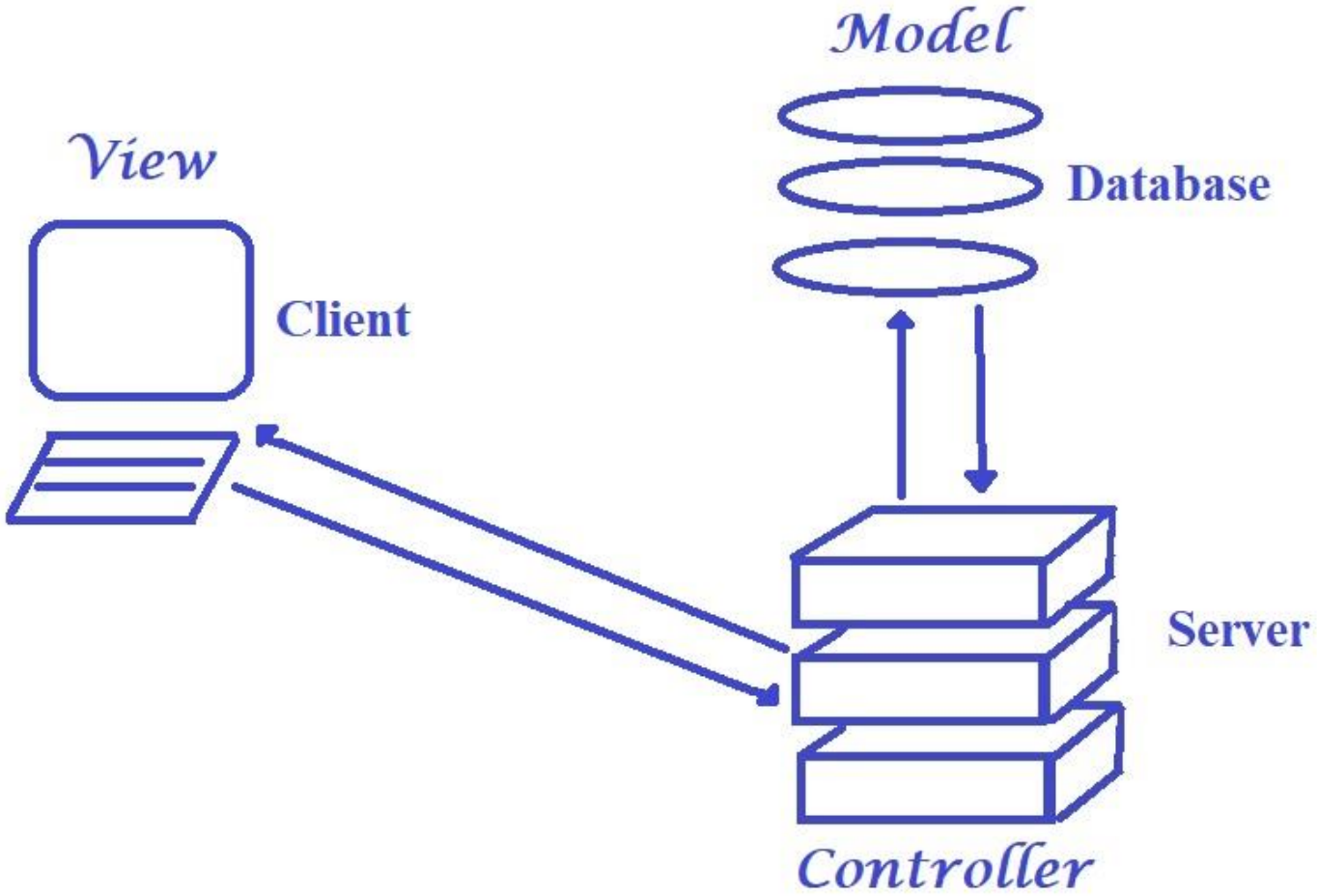
Model/View/Controller (MVC)

- Most **web frameworks** are based on the model–view–controller (MVC) pattern
  - MVC is an architectural pattern to separate the data **model** with business rules (**controller**) from the user interface (**view**).
    - This is generally considered a good practice as it modularizes code, promotes code reuse, and allows multiple interfaces to be applied.
- **Three-tier** applications are structured around three physical tiers: client, application, and database.

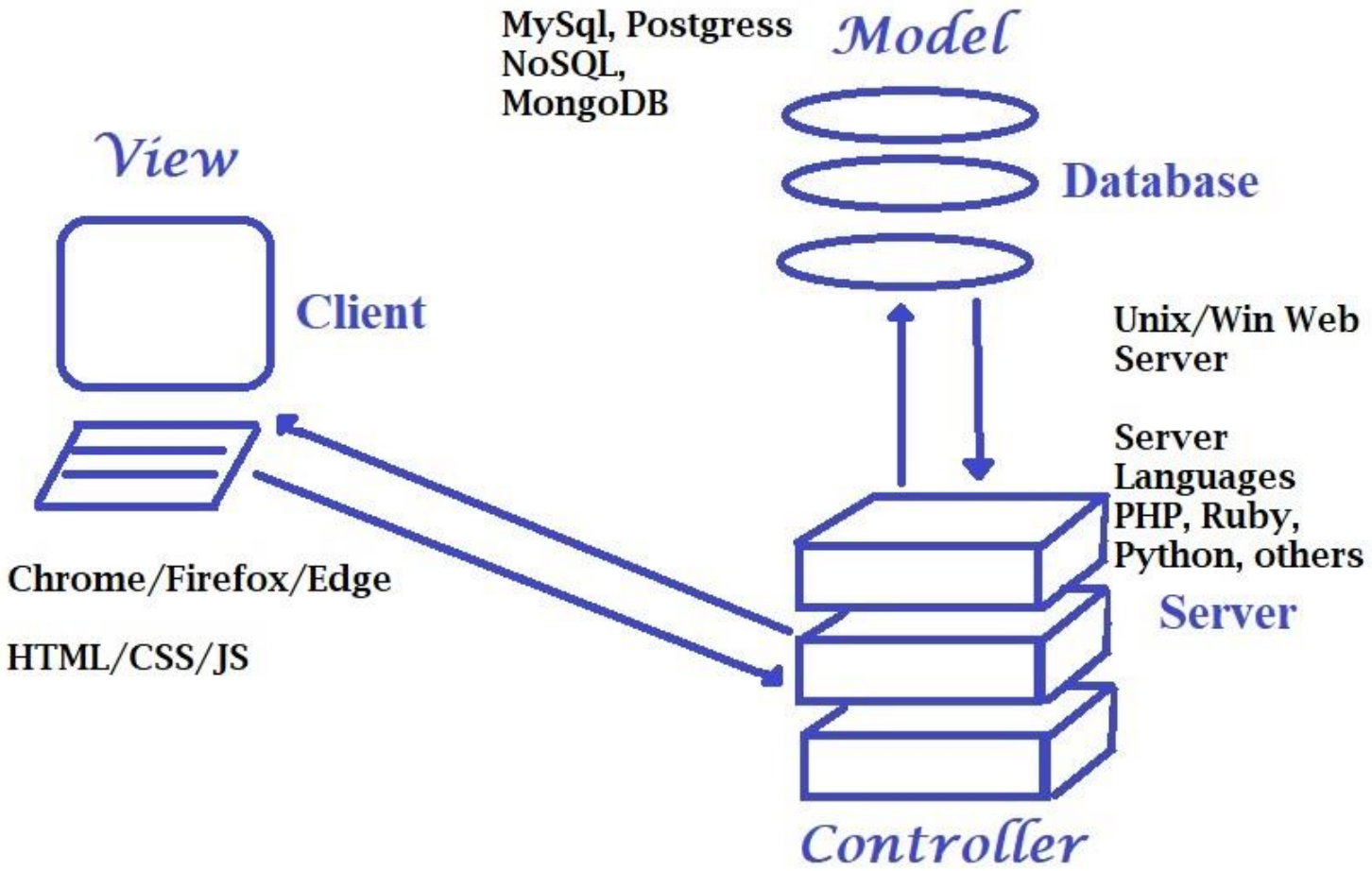
Typical Web Processing



Typical Web Processing [MVC]



Typical Web Processing [MVC + Technologies]



Web Services

- Web service is a technology that enables programs to communicate through HTTP on the Internet.
- Web services enable a program on one system to invoke a method in an object on another system.
- Standardized: You can develop and use Web services using any languages on any platform.
- There are several APIs for Web services:
  - *Simple Object Access Protocol* (SOAP), which is based on XML
  - *Representational state transfer* (REST), mostly using JSON (and also XML)

Web Services - REST

- REST – REpresentational State Transfer
  - Based on 'nouns' : e.g. products/1
  - URL is the resource identifier
  - JSON is representation of choice
  - Uses standard HTTP commands
    - GET
    - PUT
    - POST
    - DELETE
    - HEAD
  - Lower overhead than SOAP [Good for limited / mobile devices]

Uniform Resource Locator (URL)

- A URL is an address for a resource on the web, such as https://www3.cs.stonybrook.edu/~pfodor/courses/cse316.html
  - A URL consists of a protocol (https://), domain (www3.cs.stonybrook.edu), and optional path (/~pfodor/courses/cse316.html).

JSON and XML

- JSON (JavaScript Object Notation) is a text-based data storage format that is designed to be easy to read for both humans and machines.
  - JSON is generally the most common format for returning data through an API

```
{
   "firstName": "John",
   "lastName": "Smith"
}
```

- XML (Extensible Markup Language) is the second most common format for returning data through an API

```
<?xml version="1.0" encoding="UTF-8"?>
<person>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
</person>
```

The Need for REST

- REST Features
  - Separates Client and Server - Supports 'loosely coupled' applications
  - Platform and language independent
  - Not constrained by format (XML, JSON, etc)
  - Easy to use
  - Discoverable

REST with python and flask

```
pip install flask
```

- api.py:

```
import flask
app = flask.Flask(__name__)
app.config["DEBUG"] = True
@app.route('/', methods=['GET'])
def home():
    return "<h1>Test</h1>"
app.run()
```

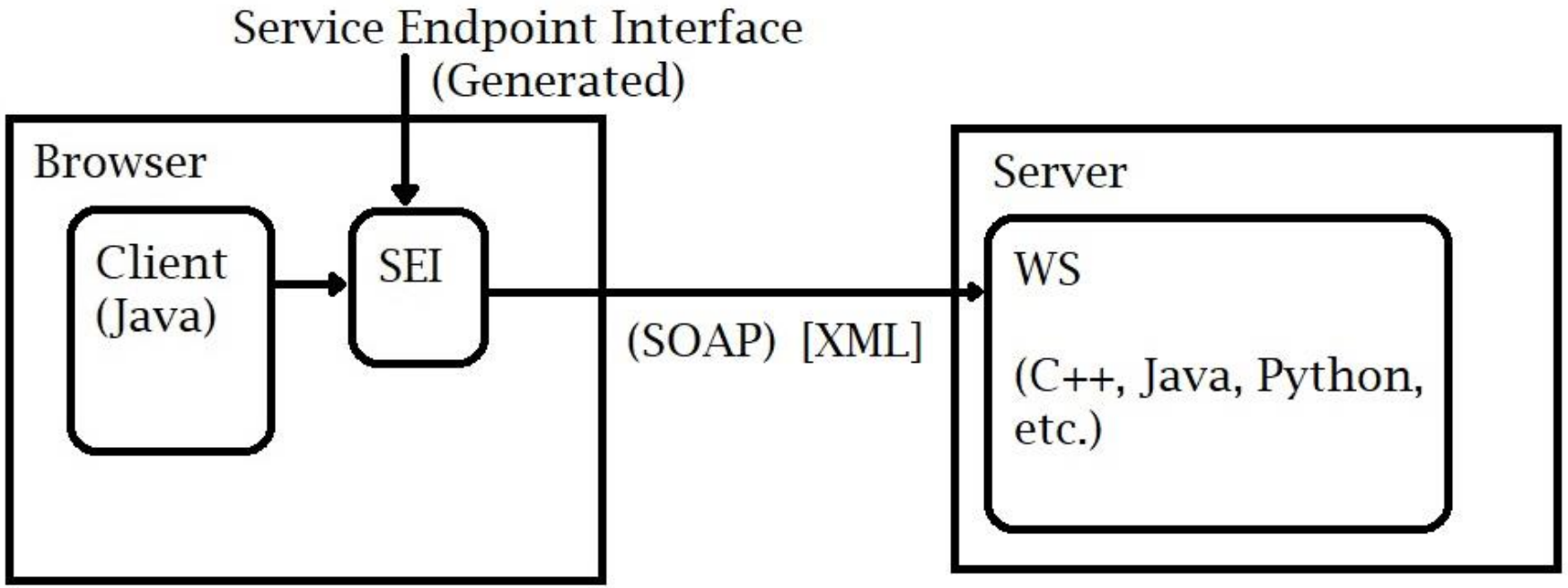
- python api.py

http://127.0.0.1:5000

Web Services – SOAP

- SOAP – Simple Object Access Protocol
  - Based on 'verbs' : e.g. getProducts/1
  - XML based messaging
  - WSDL – Web Service Description Language
    - Used to define a 'contract' describing the interface offered by a service
  - UDDI – Universal Description Discovery and Integration
  - SEI – Service Endpoint Interface
    - This is generated from the WSDL
    - Enables different client languages/environments to call the service regardless of whether these languages and environments match
  - Standard protocols (HTTP, SMTP, etc)

SOAP Web Services Model



Summary

- Creation of the web – Tim Berners Lee at CERN
  - Concept in 1989
  - First Browser/Server/Protocol in late 1990
- Early Days – 1992-1995
  - NCSA Mosaic
  - Netscape Navigator
- Commercialization – 1996-1998, .com bubble/bust – 1999-2001
- Ubiquity – 2002-present
  - New uses – [airline booking, blogs, social media, RSS, Wiki]
  - New Technologies (Web 2.0) [CSS, XML/XHTML/HTML 5/AJAX, Php…]
- Web Application Architecture
  - Client/Web Server/DB Server
  - MVC

Summary

- Web Application Architectures
  - Php
  - J2EE
  - ASP.NET
  - Node.js
- Web Services
  - REST API
  - SOAP