Web Application Development

Introduction

Because of wide spread use of internet, web based applications are becoming vital part of IT infrastructure of large organizations. For example web based employee performance management systems are used by organizations for weekly or monthly review of employees. On the other hand online course registration and examination systems can allow students to study while staying at their homes.

Web Applications

In general a web application is a piece of code running at the server which facilitates a remote user connected to web server through HTTP protocol. HTTP protocol follows stateless Request-Response communication model. Client (usually a web-browser) sends a request to Server, which sends back appropriate response or error message.
A web server is software which provides access to services to users connected to internet. These servers can provide support for many protocols used over internet or intranet like HTTP, FTP, telnet etc

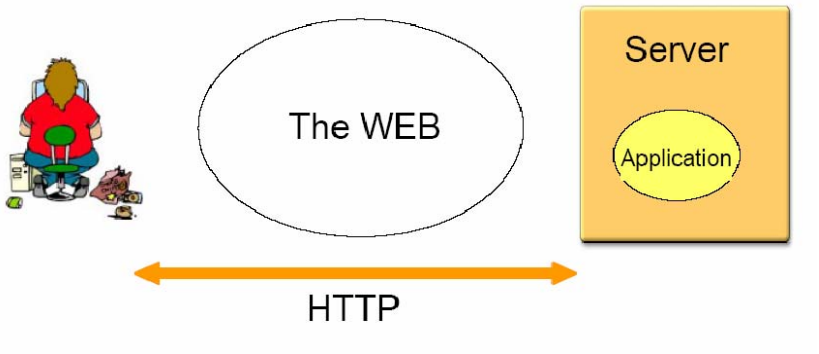

Figure 1: A typical web application

HTTP Basics

A protocol defines the method of communication between two parties. For example when we talk to our teacher the subject of discussion is studies but with our parents we normally talk about family matters. Similarly there are many different protocols used by computers to communicate with each other depending on applications. For example an Echo Server only listens to incoming name messages and sends back hello message, while HTTP protocol uses various types of request-response messages.

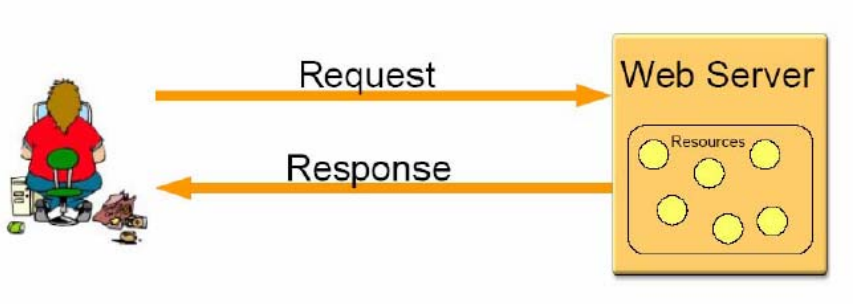

Figure 2: HTTP communication model

- HTTP is as request-response oriented protocol.
- It is a stateless protocol since there is no built-in state management between successive requests.
- Parts of an HTTP request
  - *Request Method*: It tells server what type of action client wants to be performed.
  - *URI*: Uniform Resource Indictor specifies the address of required document.
  - *Header Fields*: Option headers can be used by client to tell server extra information about request e.g. client software and content type that it understands.
  - *Body*: Contains data sent by client to server
- Other request headers like FROM (email of the person responsible for request) and VIA (used by gateways and proxies to show intermediate sites the request passes) can also be used.

```
GET /index.html HTTP/1.1          request line

Host: java.sun.com                request headers
User-Agent: Mozilla/4.5 [en]
Accept: image/gif, image/jpeg, image/pjpeg, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

                                  optional request body
Request parameters etc
```

Figure 3: HTTP request example

- Request can also contain addition information in form of request parameters
  - In URL as query string e.g. http://www.gmail.com/register?name=ali&state=punjab
  - As part of request body (see Figure 1)

- Parts of HTTP response
  - *Result Code*: A numeric status code and its description.
  - *Header Fields*: Servers use these fields to tell client about server information like configurations and software etc.
  - *Body*: Data sent by server as part of response.

```
HTTP/1.1 200 OK                          status line

Last-Modified: Mon, Aug 4 2003 22:10:40 GMT
Date: Wed, Aug 8 2003 14:23:35 GMT
Status: 200
Content-Type: text/html
Content-Length: 59                  response headers

<html>                        optional response body
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```

Figure 4: HTTP response example

- HTTP codes fall into five general categories
  - 100-199
    - Codes in the 100s are informational, indicating that the client should respond with some other action.
    - 100: Continue with partial request.
  - 200-299
    - Values in the 200s signify that the request was successful.
    - 200: Means every thing is fine.
  - 300-399
    - Values in the 300s are used for files that have moved and usually include a Location header indicating the new address.
    - 300: Document requested can be found several places; they'll be listed in the returned document.
  - 400-499
    - Values in the 400s indicate an error by the client.
    - 404: Indicates that the requested resource is not available.
    - 401: Indicates that the request requires HTTP authentication.
    - 403: Indicates that access to the requested resource has been denied.
  - 500-599
    - Codes in the 500s signify an error by the server.
    - 503: Indicates that the HTTP server is temporarily overloaded and unable to handle the request.

Server Side Programming

Web server pages can be either static pages or dynamic pages. A static web page is simple HTML (Hyper Text Transfer Language) file. When a client requests an HTML page the server simple sends back response with the required page.

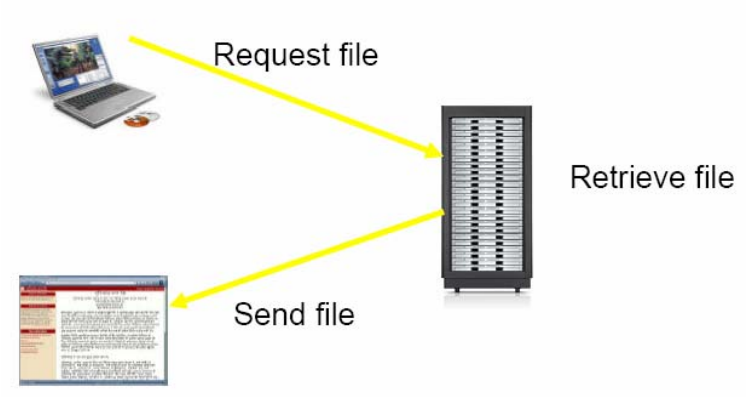

Figure 5: Static web page request and response

While in case of dynamic web pages server consists of a running application which generates HTML web pages according to specific requests coming from client. These dynamically generated web pages are sent back to client with the response.

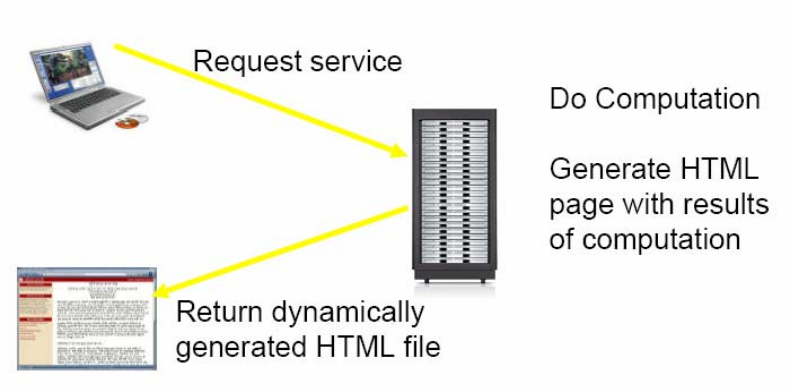

Figure 6: Dynamic web page request and response

We need to create dynamic web pages when the content of site changes frequently and client specific response is required. Some of the scenarios are listed below

- The web page is based on data submitted by the user. e.g. results page from search engines and order confirmation pages at on line stores.
- The Web page is derived from data that changes frequently. e.g. a weather report or news headlines page.

- The Web page uses information from databases or other server-side resources. e.g. an e-commerce site could use a servlet to build a Web page that lists the current price and availability of each item that is for sale.

Server side programming involves

- Using technologies for developing web pages that include dynamic content.
- Developing web based applications which can produce web pages that contain information that is connection-dependent or time-dependent.

Dynamic web content development technologies have evolved through time in speed, security, ease of use and complexity. Initially C based CGI was used to write server side programs which were to provide limited but fast services to users. Then template based technologies like ASP and PHP were introduced which allowed easy of use for designing complex web pages. Sun Java introduced Servlets and JSP that provided more speed and security as well as better tools for web page creation.

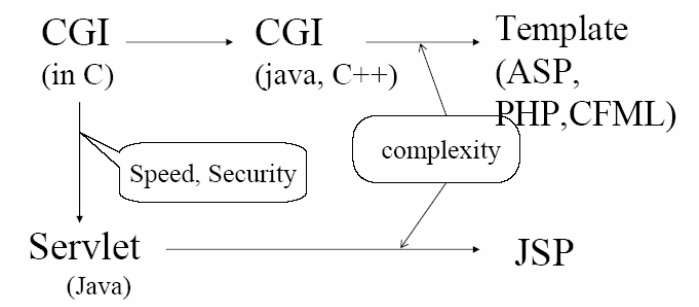

Figure 7: Dynamic web content technologies evolution

Normally web applications are partitioned into three layers. Each layer performs a specific functionality which should not be mixed with other layers. Layers are isolated from each other to reduce coupling between them but they provide interfaces to communicate with each other.

- *Presentation Layer*: Provides user interface for client to interact with the application. This is the only part of application visible to client.
- *Business Layer*: The business or service layer implements the actual business logic or functionality of the application. For example in case of online shopping systems this layer handles transaction management.
- *Data Layer*: This layer consists of objects that represent real-world business objects such as an *Order, OrderLineItem, Product,* and so on.

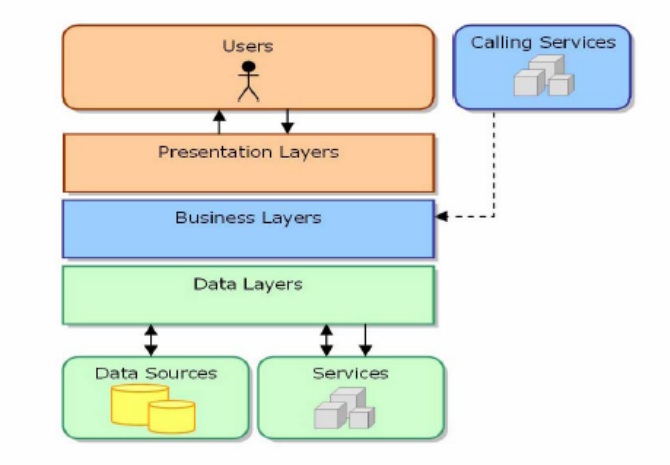

Figure 8: Simplified view of a web application and its layers

Java technologies for web application development include Java Servlets, Java Server Pages, EJB, Java Server Faces and etc.

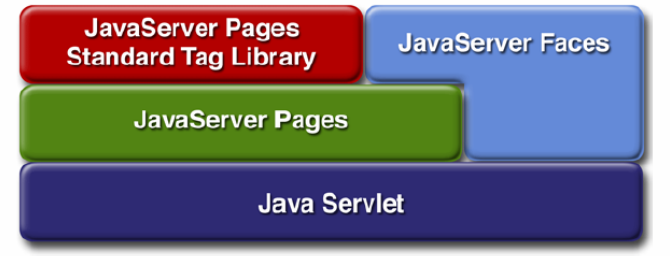

Figure 9: Java web application technologies (presentation/web tier)