Desarrollo de Aplicaciones Web

Tema 1. Aplicaciones Web

Contenido

- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

Contenido

- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

Introducción

- Página Web

Introducción

- Aplicación web = Página web+funcionalidad dinámica servidor
  - HTML (o XHTML), CSS, Imágenes, sonidos, etc.
  - Programación en el cliente (Javascript), para dar funcionalidad en el lado del cliente
  - Programación en el servidor (PHP, Servlets, JSP, ASP.NET, etc), que permite generar el contenido HTML de forma dinámica mediante una lógica de negocio

  Definición: Una aplicación Web es un sitio Web donde la navegación a través del sitio y la entrada de datos por parte de un usuario requieren de la ejecución de una "lógica del negocio" en el servidor, para generar los recursos que serán visualizados (renderizados) en un navegador.

Introducción

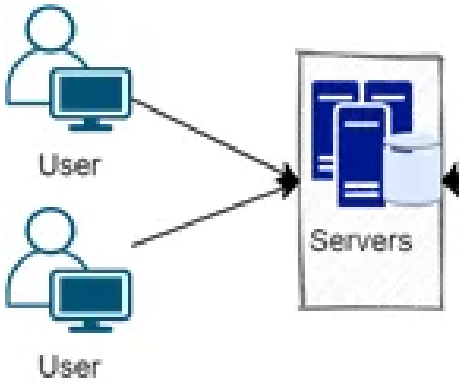
- Aplicación Web
  - basada en el World Wide Web (WWW)
- Características
  - Arquitectura Cliente/Servidor
  - Protocolo: HTTP (HyperText Transfer Protocol)
  - Tecnologías estándar de la Web: HTML, CSS y JS
  - Identificación de Recursos: URL (Uniform Resource Locator)
  - Cliente : Navegador o Browser (Internet Explorer, Firefox, Chrome, Safari, Opera, etc…), capaz de interpretar HTML, CSS y JS
  - Servidor: servidor web (Apache HTTP, IIS, etc) o servidor de aplicaciones (Glassfish, Tomcat, etc…) y lenguajes o frameworks (PHP, Servlets, JSP, ASP.NET, Python, Django, …) capaces de generar contenido HTML de forma dinámica

Contenido

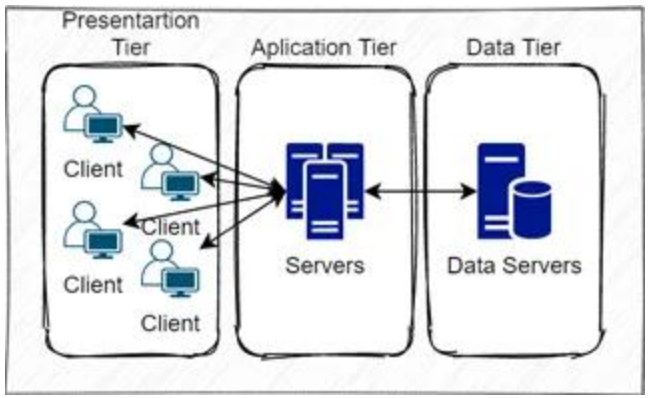
- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

Arquitectura Cliente/Servidor

- Modelo en dos capas

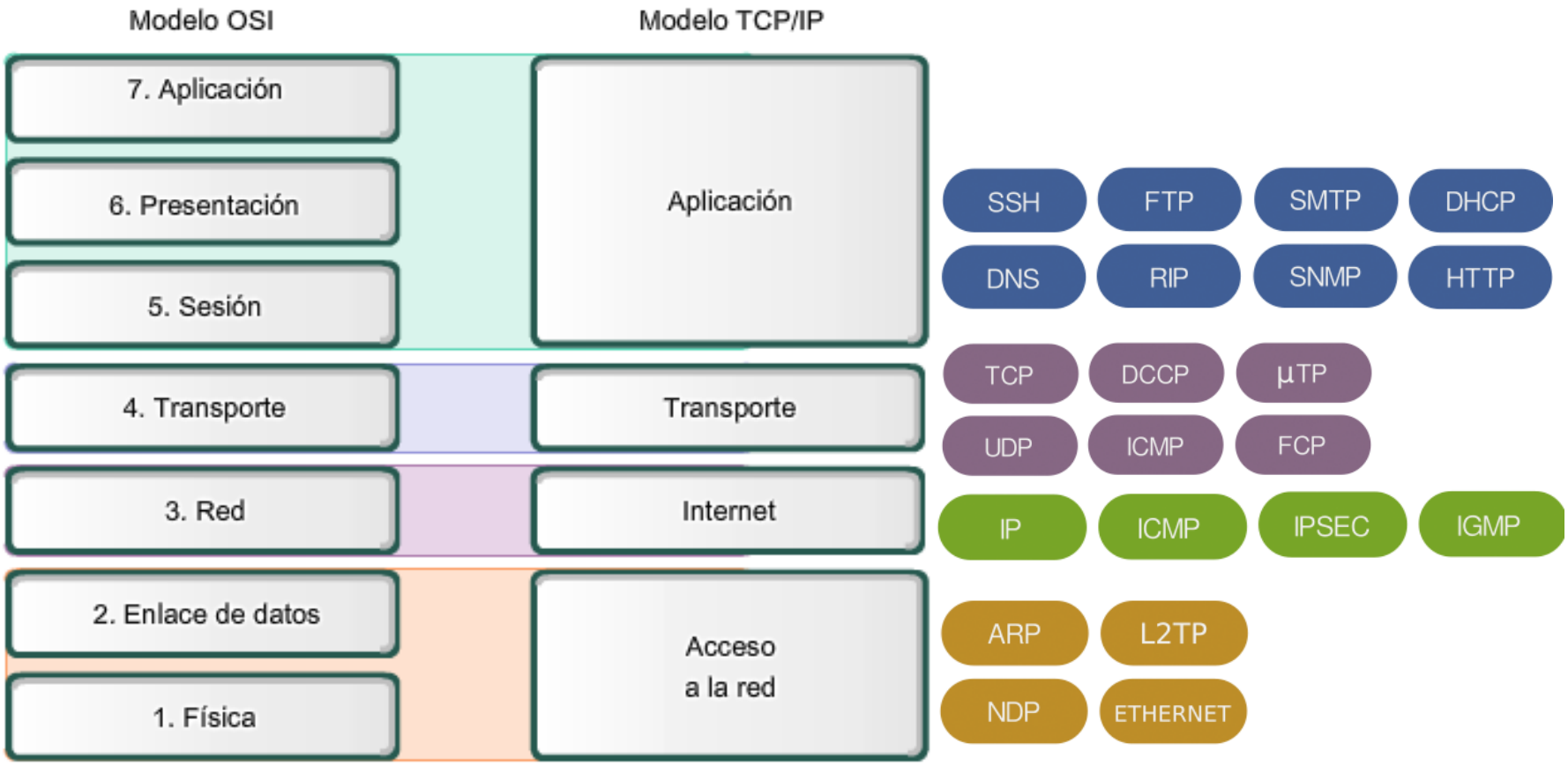


- Modelo en tres capas



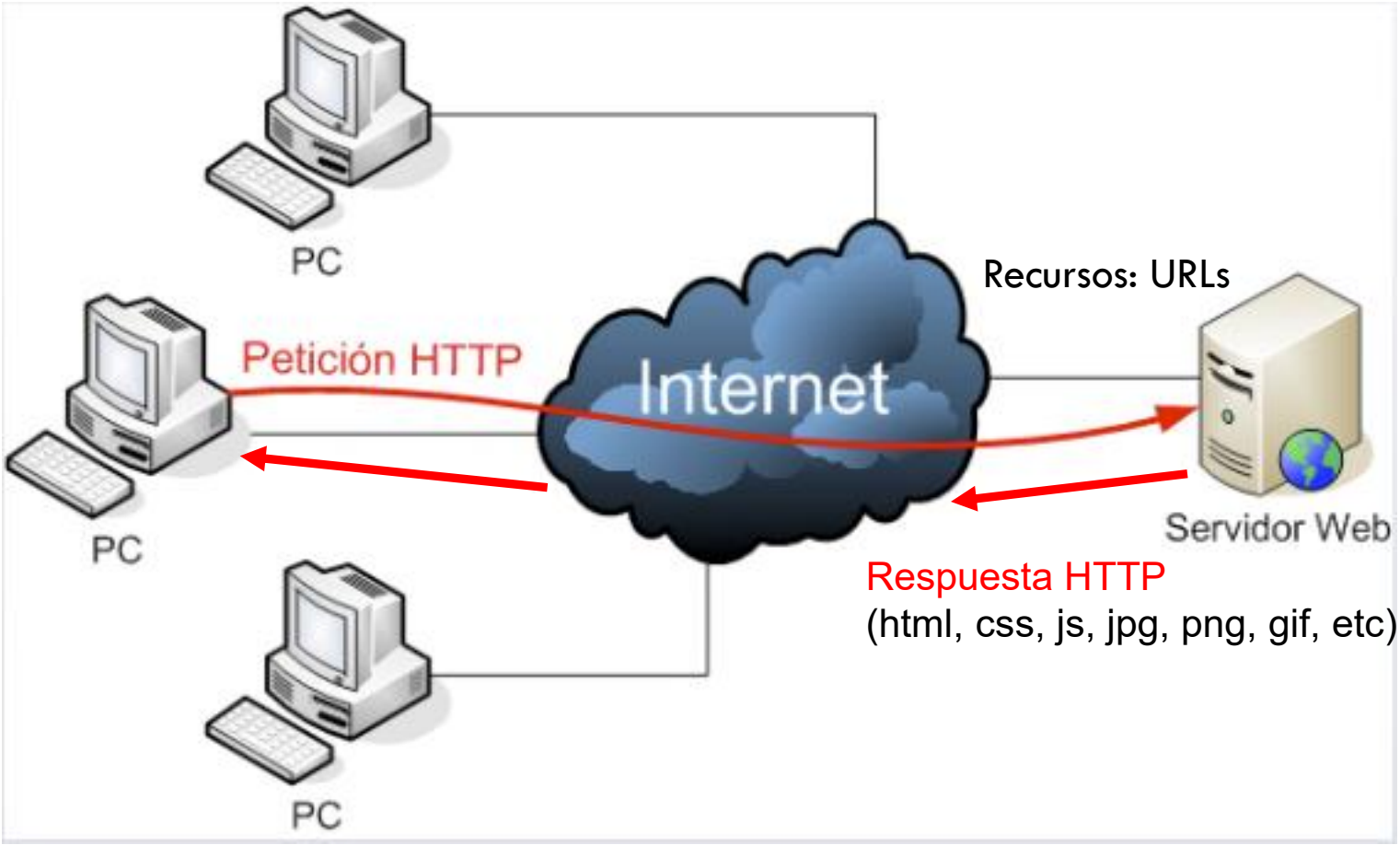
Arquitectura Cliente/Servidor

- Protocolos de Internet o Protocolos TCP/IP



Arquitectura Cliente/Servidor

- Arquitectura Web



Contenido

- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

Protocolo HTTP

- HTTP (HyperText Transfer Protocol)
  - Protocolo para transferencia mediante hipertexto

https://developer.mozilla.org/es/docs/Web/HTTP

- Basado en petición/respuesta

Protocolo HTTP

1. Socket on the server side is created (http://online-shop.com:8080)
2. Server socket waits for connection
3. Socket on the client side is created
4. Connect to the server socket
5. Send request
6. Handle request and send response
7. Closing the connection
8. Response parsed on the client side

Protocolo HTTP

- Peticiones del Protocolo HTTP
  - HEAD
  - GET
  - POST
  - PUT
  - DELETE
  - TRACE
  - OPTIONS
  - CONNECT
  - PATCH
  - SEARCH
  - COPY
  - LOCK
  - UNLOCK
  - MOVE
  - MKCOL
  - PROPFIND
  - PROPPATCH
  - MERGE
  - UPDATE
  - LABEL

Para más información:

https://developer.mozilla.org/es/docs/Web/HTTP/Methods

Protocolo HTTP

- Principales peticiones
  - **GET**: Realiza la petición de un determinado recurso, pudiendo añadir información como parámetros (Query parameters o Path parameters)

    http://www.host.org:port/path/resource

    http://www.host.org:port/path/resource?name=pepe&edad=18

    http://www.host.org:port/path/resource/pepe/18
  - **POST**: Realiza la petición de un determinado recurso, añadiendo un bloque adicional de información en el cuerpo (body)

```
<h3>Ejemplo formulario HTML</h3>
<form  action="http://www.host.es:port/resource“ method=“POST">
Nombre: <input type="text" name=“nombre"><br>
<input type="submit“ value=“Enviar">
</form>
```

Protocolo HTTP

- Estructura de las peticiones
  - Línea de petición (método: GET, POST, ... + URL recurso)

    Por ejemplo, un recurso estático como un fichero HTML, CSS, o JS; o un recurso dinámico como un PHP, un Servlet o un JSP que genera de forma dinámica el contenido estático (HTML)
  - Líneas de Cabeceras (Información adicional sobre la petición)

    Navegador, idioma, contenido aceptado, juego de caracteres, información de autorización, …
  - Cuerpo de la petición

    Vacío para peticiones GET y datos enviados desde peticiones POST, PUT, etc.

    Por ejemplo, datos enviados desde un formulario (Form Data) o datos JSON enviados desde un cliente REST

Protocolo HTTP

- Ejemplo de petición

```
GET /search?q=Web HTTP/1.1

Host: www.google.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.7.2)
Accept: text/xml,application/xml,application/xhtml+xml,
    text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: da,en-us;q=0.8,en;q=0.5,sw;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://www.google.com/
```

Protocolo HTTP

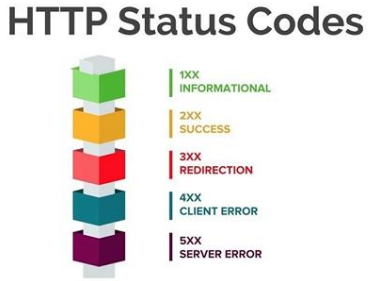
- Estructura de las Respuestas
  - Línea de estado de la respuesta

    Códigos de estado 1xx, 2xx, 3xx, 4xx ó 5xx
  - Líneas de Cabeceras con información del servidor

    Tiempo de respuesta, información del servidor, fecha última modificación, …
  - Cuerpo de la respuesta

    recurso solicitado, puede ser texto plano, como código HTML, código javascript, código CSS, o binario, como una imagen (jpg, gif, etc).

Protocolo HTTP



- Códigos de Estado
  - 1xx: Informan al navegador de algunas acciones que se van a realizar
  - 2xx: Indican que la petición se ha recibido, procesado y respondido correctamente
  - 3xx: Indican que el navegador debe realizar alguna acción adicional para completar la petición
  - 4xx: Indican que se ha producido un error cuyo responsable es el navegador
  - 5xx: Indican que se ha producido un error cuyo responsable es el servidor

Más información en:

https://developer.mozilla.org/es/docs/Web/HTTP/Status

Protocolo HTTP

- Ejemplo de respuesta

```
HTTP/1.1 200 OK
Date: Fri, 17 Sep 2009 07:59:01 GMT
Server: Apache/2.0.50 (Unix) mod_perl/1.99_10 Perl/v5.8.4
    mod_ssl/2.0.50 OpenSSL/0.9.7d DAV/2 PHP/4.3.8 mod_bigwig/2.1-3
Last-Modified: Tue, 24 Feb 2009 08:32:26 GMT
ETag: "ec002-afa-fd67ba80"
Accept-Ranges: bytes
Content-Length: 2810
Content-Type: text/html

<!DOCTYPE HTML>
<html>
...
</html>
```

Protocolo HTTP

- Ejercicio: Protocolo HTTP en estado puro

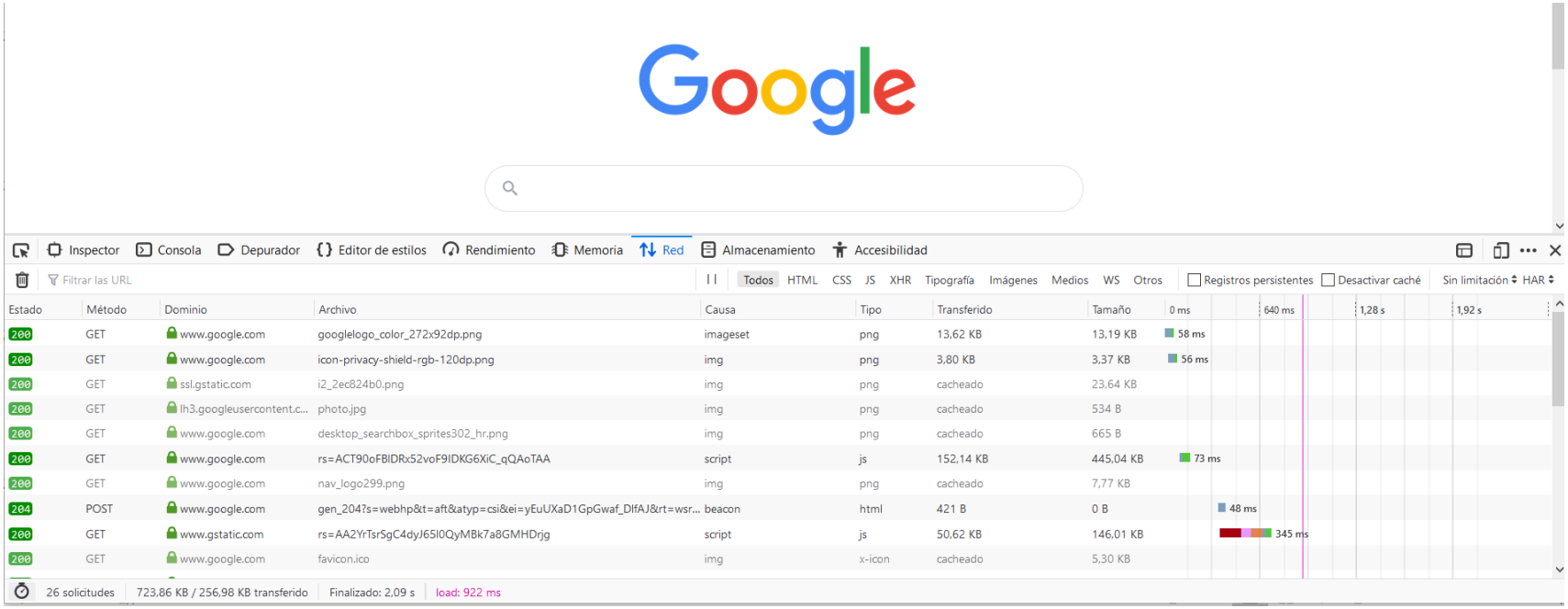
  Analicemos las peticiones y respuestas que se producen de forman transparente al usuario en un navegador cuando solicita una página web.

  - En Firefox: Menú + Desarrollador Web + Red
  - En Chrome: Menú + Más herramientas + Htas. Desarrolladores + Network
  - En Edge: Menú + Más Herramientas + Herramientas de Desarrollo + Red

Protocolo HTTP

- Ejercicio: Protocolo HTTP en estado puro

Acceder, por ejemplo, desde Firefox, a www.google.com y comprobar el tráfico de peticiones que se producen. Analiza alguno de los resultados.



Protocolo HTTP

- Limitaciones del protocolo
  - Ausencia de estado: no mantiene traza o secuencia de acciones del usuario (se requiere alguna técnica que permita gestionar esta información de cada usuario)
  - No dispone de mecanismo propio de seguridad (se requiere hacer uso de tecnología adicional (SSL – Secure Sockets Layer) para resolver este problema)

Contenido

- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

Tecnologías en el cliente

- Cliente: Navegadores (Browsers)
  - Aplicación que permite realizar peticiones (http) a un servidor e interpretar la respuesta.
  - Interpreta HTML, CSS y Javascript
  - Ampliación de funcionalidad:
    - Aplicaciones auxiliares (helper applications)
    - Conectores o Complementos (plugins, activeX)
  - Configuración
    - Establecer la configuración de comunicación (Proxy, etc)
    - Establecer la seguridad.
    - Habilitar o deshabilitar aspectos para la navegación (javascript, cookies, etc...)

Tecnologías en el cliente

- Comparativa de Navegadores

Mosaic

Netscape

Internet Explorer

Chrome

FireFox

Safari

Internet Explorer
Microsoft EDGE

Opera

Brave

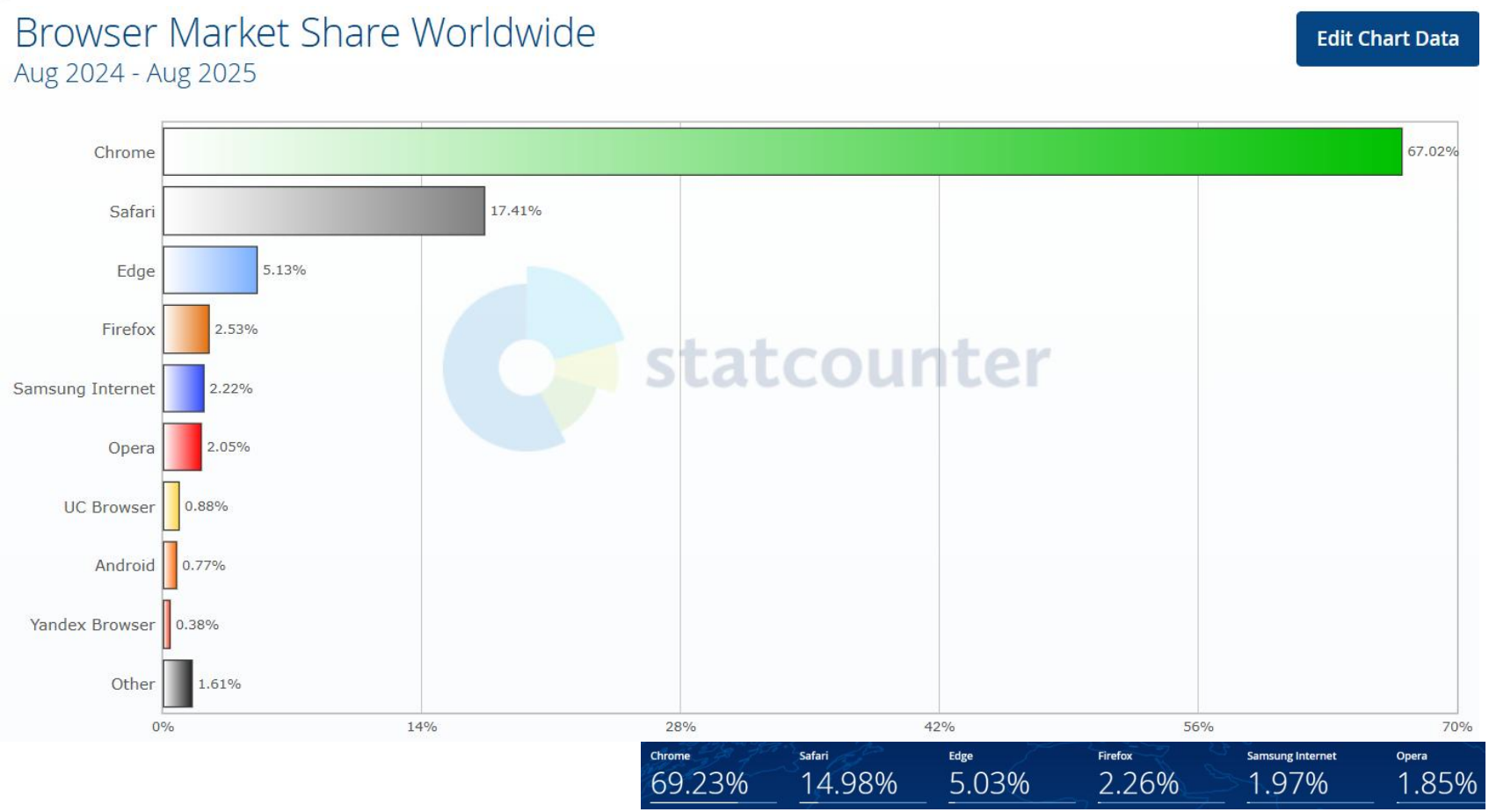
http://es.wikipedia.org/wiki/Comparativa_de_navegadores_web

Navegadores

2025	Chrome	Edge	Firefox	Safari	Opera
August	80.4 %	11.2 %	3.5 %	3.4 %	0.9 %
July	80.5 %	11.3 %	3.6 %	3.3 %	0.8 %
June	79.8 %	11.4 %	3.6 %	3.7 %	0.9 %
May	79.1 %	11.6 %	3.7 %	4.0 %	1.0 %
April	79.1 %	11.4 %	3.8 %	4.2 %	0.8 %
March	78.5 %	11.6 %	3.8 %	4.2 %	0.8 %
February	78.4 %	11.7 %	3.9 %	4.3 %	0.9 %
January	79.4 %	10.7 %	3.9 %	3.9 %	1.4 %
2024	**Chrome**	**Edge**	**Firefox**	**Safari**	**Opera**
December	79.6 %	10.5 %	3.9 %	3.9 %	1.4 %
November	78.5 %	11.1 %	4.0 %	4.0 %	1.6 %
October	77.9 %	11.2 %	4.2 %	4.1 %	1.6 %
September	78.6 %	11.1 %	4.0 %	3.6 %	1.6 %

http://www.w3schools.com/browsers/browsers_stats.asp

Navegadores



https://gs.statcounter.com

Sistemas Operativos

2025	Win11	Win10	Win8	Win7	Linux	Mac	ChromeOS	Mobile
August	43.0	24.3	0.3	0.8	3.9	9.3	0.4	17.9
July	44.0	25.8	0.3	0.9	4.1	6.8	0.3	17.7
June	43.6	26.1	0.3	0.8	3.9	7.7	0.4	17.3
May	42.0	27.4	0.3	0.8	3.9	8.0	0.7	17.1
April	40.7	27.4	0.3	0.8	3.8	8.1	0.6	18.2
March	39.8	28.2	0.3	0.8	3.9	7.9	0.6	18.1
February	39.6	28.7	0.4	0.9	3.9	7.8	0.6	17.7
January	38.1	31.3	0.4	0.9	4.0	7.7	0.6	16.7

2024	Win11	Win10	Win8	Win7	Linux	Mac	ChromeOS	Mobile
December	37.3	30.8	0.3	1.0	3.9	7.3	0.6	18.3
November	36.0	32.5	0.4	1.0	3.6	7.4	0.6	17.8
October	35.4	32.8	0.4	0.9	3.7	7.9	0.6	16.3
September	35.8	32.1	0.4	1.0	3.8	7.6	0.6	16.5

http://www.w3schools.com/browsers/browsers_os.asp

Tecnologías del Cliente

- Tecnologías estándar de la web
  - HTML
    - Contenido / Estructura
  - CSS
    - Estilo / Presentación
  - JavaScript
    - Comportamiento

Contenido

- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

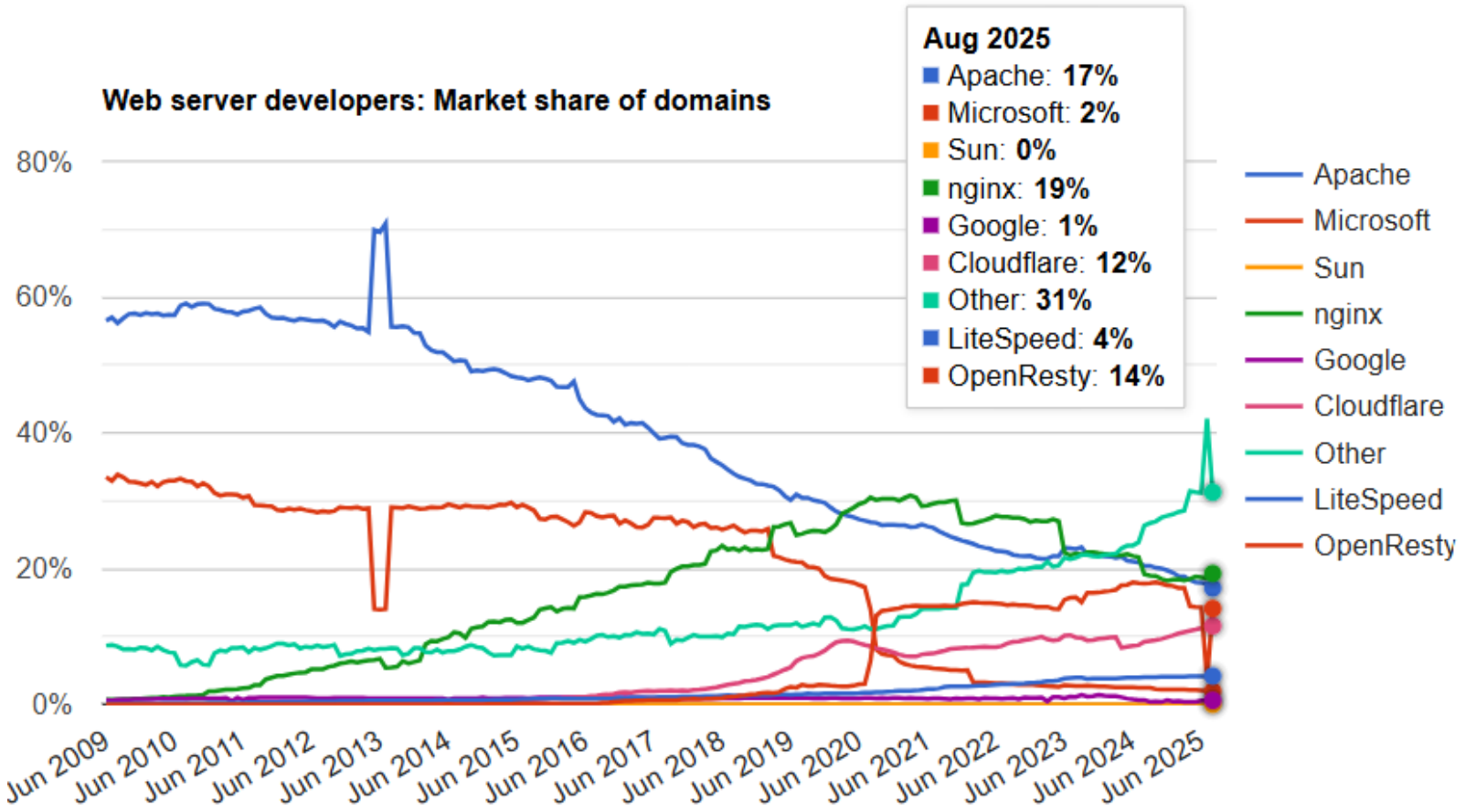
Tecnologías en el Servidor

- Servidores Web y Servidores de Aplicaciones
  - Aplicación que atiende peticiones HTTP, enviando, como respuesta, documentos HTML, CSS, JS, imágenes, etc
  - En las aplicaciones Web se require que el servidor ejecute código, para generar de forma dinámica la respuesta y, entre otras acciones, conexiones con bases de datos.
  - Aspectos de interés
    - Funcionalidad
    - Seguridad
    - Administración y configuración
    - Soporte

Tecnologías en el Servidor

- Principales Servidores web
  - Apache HTTP Server
  - Microsoft IIS
  - NGINX
  - Lighttpd

Tecnologías en el Servidor



https://www.netcraft.com/blog/august-2025-web-server-survey

Tecnologías en el Servidor

- Servidores de Aplicaciones para tecnologías Java
  - Apache TomCat

  - Eclipse GlassFish

  - WildFly

  - WebLogic

  - WebSphere

  - Jetty

Tecnologías en el Servidor

- Lenguajes de programación y Frameworks
  - Tecnología Java: Servlets y JSP (Java Server Pages)
  - PHP
  - Python
  - ASP.NET
  - etc

  - Spring
  - Django
  - Express, Flask, etc…

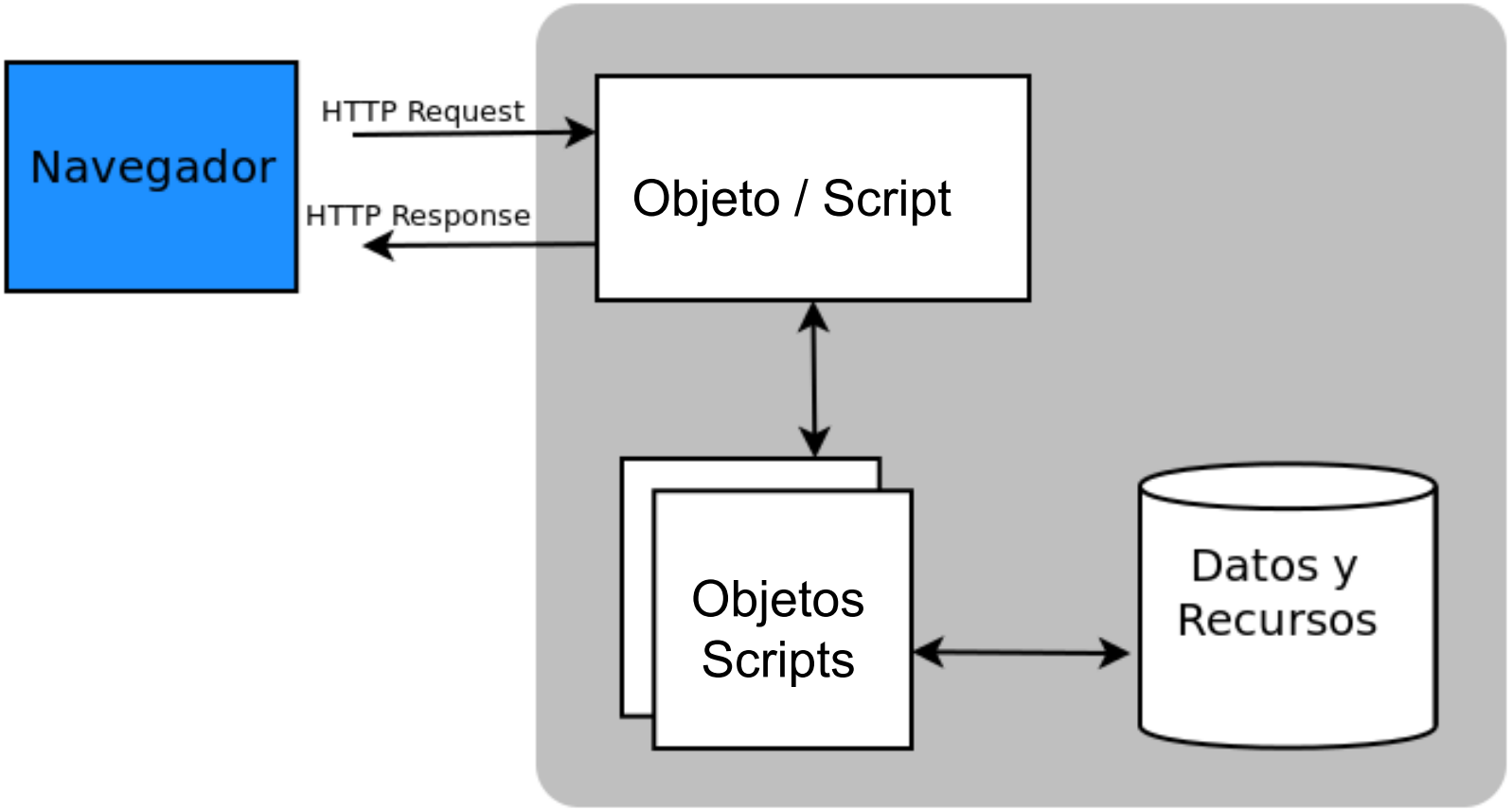
Tecnologías en el Servidor

- Monitorización de tecnologías

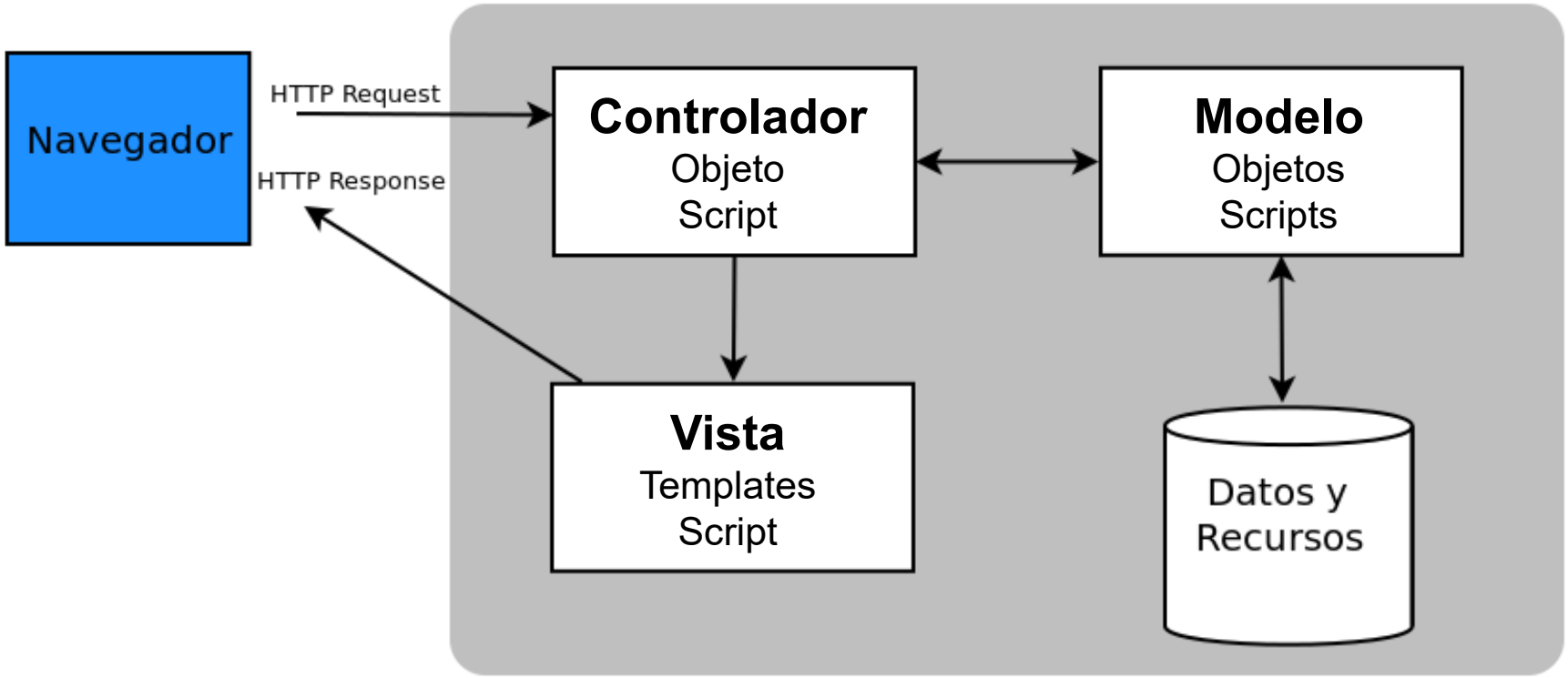
Contenido

- Introducción
- Arquitectura Cliente/Servidor
- Protocolo HTTP
- Tecnologías del Cliente
- Tecnologías del Servidor
- Patrón Model-View-Controller

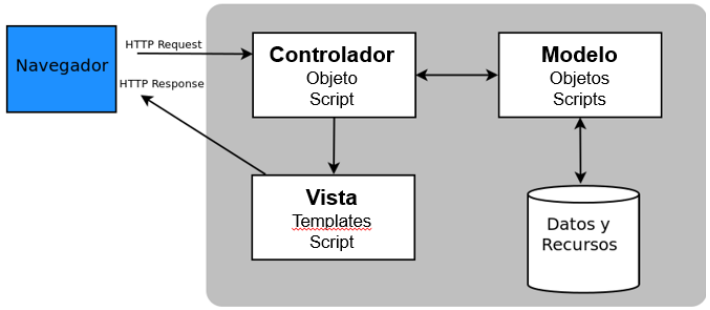
Modelo Tradicional



Patrón MVC: Model View Controller



Patrón MVC: Model View Controller



- Ventajas:
  - Mejor separación de competencias (conceptos)
  - Separación del desarrollo en capas: lógica de negocio, capa de datos, de presentación.
  - Mejor control de la Navegación (concentrada en controladores).
  - Facilidad de mantenimiento
  - Facilidad de Ampliación

Desarrollo de Aplicaciones Web

Tema 1. Aplicaciones Web