UT 4: WWW y protocolo HTTP

2ºDAW – Despliegue Aplicaciones Web

WWW (World Wide Web)

- ✓ Servicio de distribución de información.

- ✓ Acceso a millones de recursos electrónicos y aplicaciones aplicaciones distribuidos en servidores servidores por todo Internet.
- ✓ Identificados y localizados por direcciones (URIs o URLs).

- ✓ Conectados entre sí a través de hiperenlaces (o hipervínculos)

W3C y estándares web

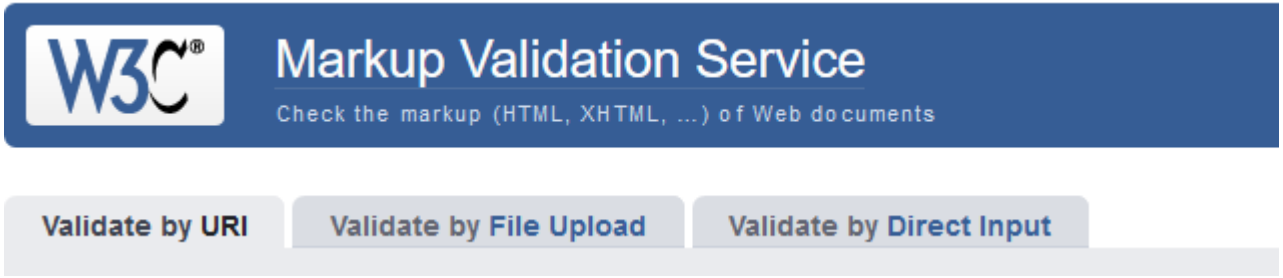
- ✓ **WWW** Desarrollada por el CERN (Centro Europeo de Investigación Nuclear) en 1989.

- ✓ **W3C(World Wide Web Consortium)**

  http://www.w3.org/
  http://www.w3c.es/

Comunidad internacional que controla el desarrollo de la WWW.
Desarrolla estándares web, por ejemplo XHTML, CSS y XML

Validación webs http://validator.w3.org/



Páginas, sitios y aplicaciones Web

Página web

- ✓ Documento hipermedia o conjunto de información electrónica relacionada (texto, audio, imágenes, video, etc.) que normalmente contiene hiperenlaces a otras paginas web o recursos.
- ✓ Escrita en lenguajes lenguajes lenguajes que son interpretados y/o ejecutados por los navegadores (XHTML, CSS, CSS, Javascript, …)
- ✓ Contenidos estático y dinámico

Sitio web

- ✓ Conjunto de paginas web relacionadas y accesibles a partir de un mismo nombre de dominio DNS.
- ✓ El conjunto de sitios web de Internet constituyen la WWW

Web estáticas vs Web dinámicas

Páginas web estáticas: almacenadas en su forma definitiva, tal y como se crearon, y su contenido no varía. Son útiles para mostrar una información concreta, y mostrarán esa misma información cada vez que se carguen.

Páginas web dinámicas: contenido cambia en función de diversas variables, como puede ser el navegador que estás usando, el usuario con el que te has identificado, o las acciones que has efectuado con anterioridad.

Dos tipos web dinámicas:

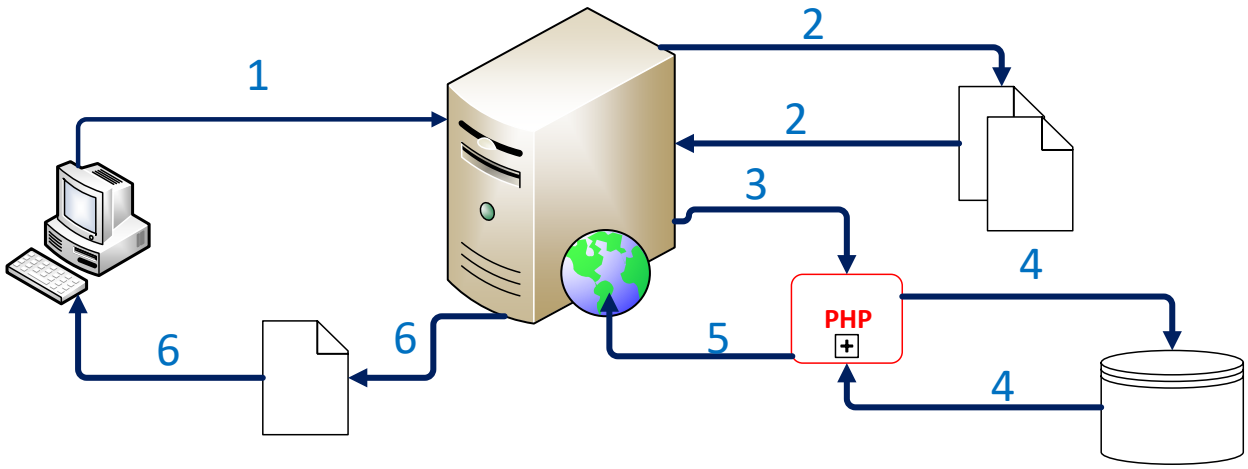
- Aquellas que incluyen código que ejecuta el navegador. El código ejecutable (normalmente JavaScript) se incluye dentro del HTML y se descarga junto con la página. Cuando el navegador muestra la página en pantalla, ejecuta el código embebido.

- Aquellas en que HTML se forma como resultado de la ejecución de un programa (ejecución tiene lugar en el servidor web). Páginas con extensiones .php, .asp, .jsp, .cgi o .aspx. Contenido que se descarga al navegador es similar al de una página web estática.

Web estáticas vs Web dinámicas

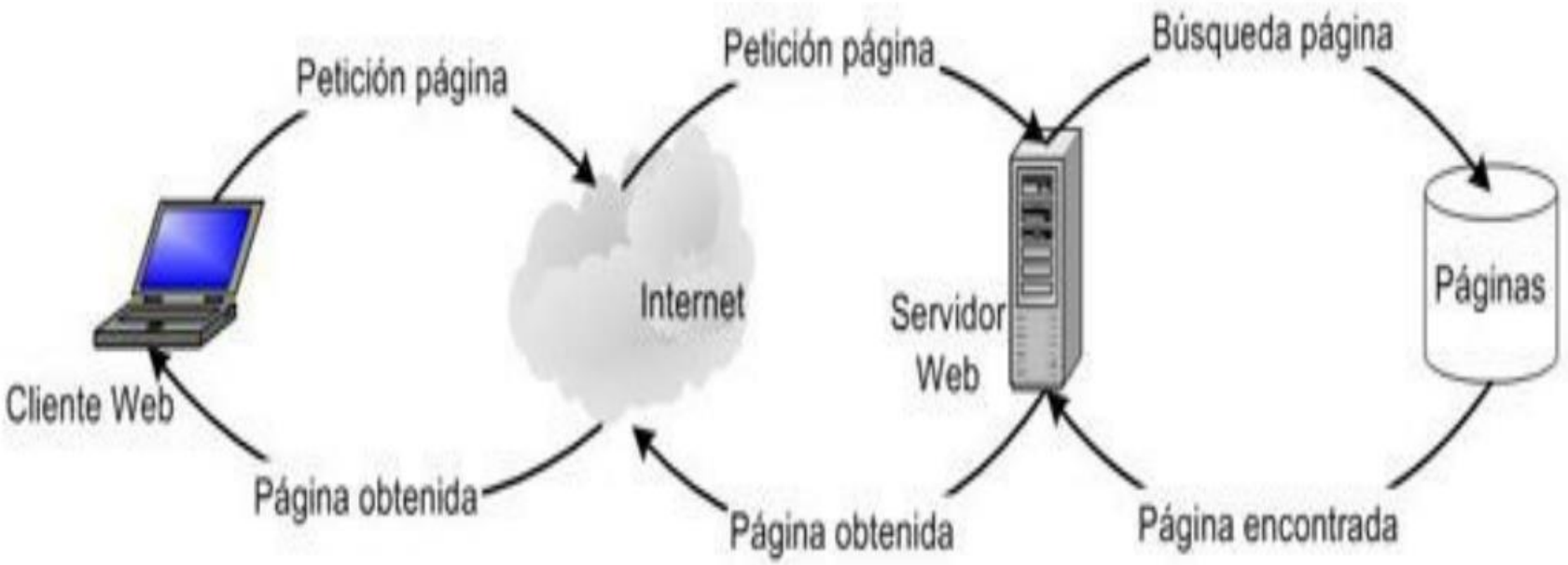
Funcionamiento:

1. Cliente web solicita a un servidor web una página web.
2. El servidor busca esa página y la recupera.
3. En el caso de que se trate de una página web dinámica (contenido debe ejecutarse para generar HTML se enviará al cliente), el servidor web contacta con el módulo responsable de ejecutar el código y se lo envía.
4. Como parte del proceso de ejecución, puede ser necesario obtener información de algún repositorio (normalmente una base de datos).
5. El resultado de la ejecución será una página en formato HTML, similar a cualquier otra página web no dinámica.
6. El servidor web envía el resultado obtenido al cliente web, que la procesa y muestra en pantalla.



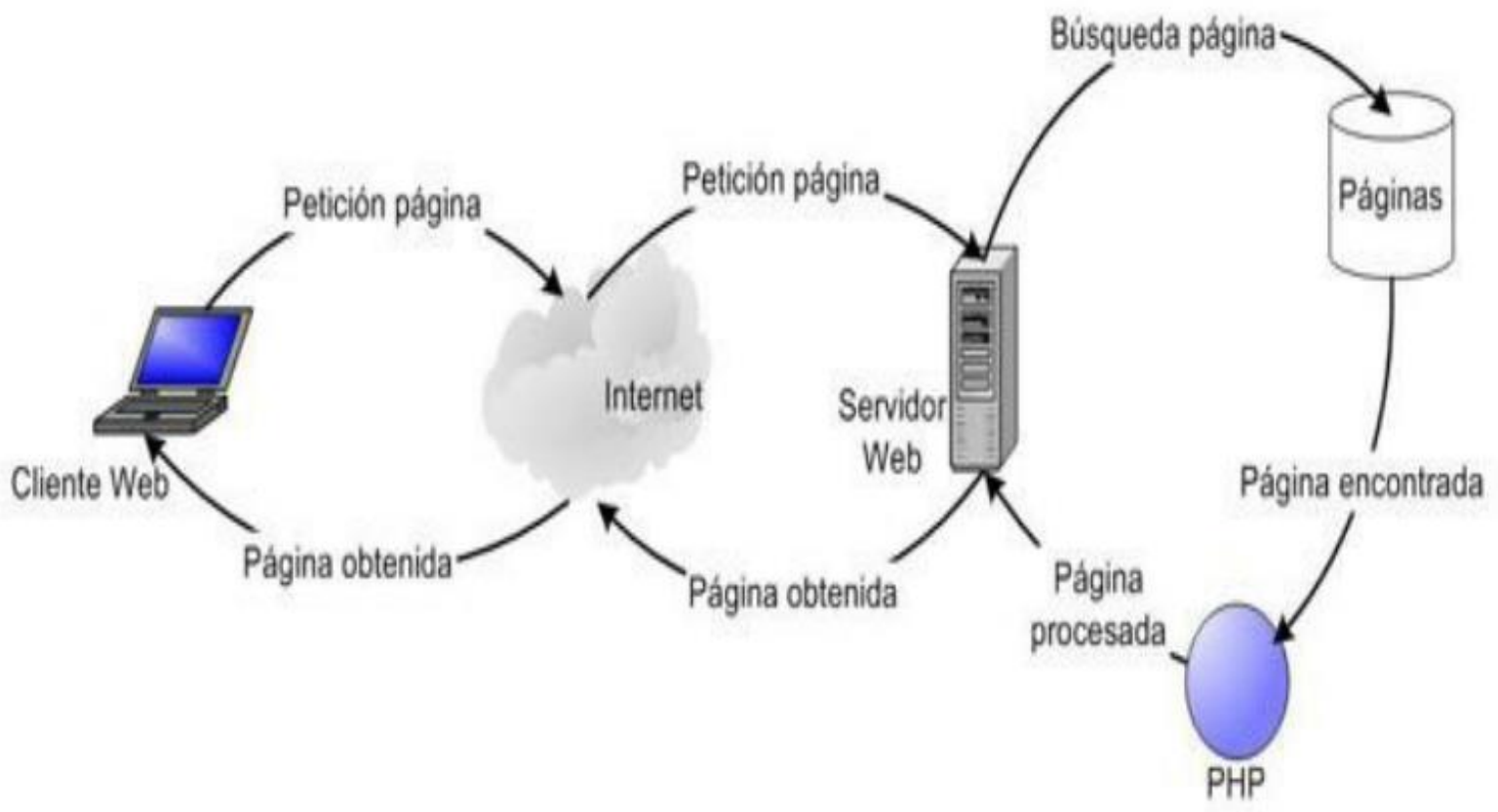
Web estáticas vs Web dinámicas

Web estática



Web estáticas vs Web dinámicas

Web dinámica



Web estáticas vs Web dinámicas

Ventajas web estáticas:

- ✓ No es necesario saber programar para crear un sitio que utilice únicamente páginas web estáticas. Simplemente habría que conocer HTML/XHTML y CSS, e incluso esto no sería indispensable se podría utilizar algún programa de diseño web para generarlas.

- ✓ Sólo necesitan un servidor web que se comunique con navegador, no necesita módulos para procesar una web dinámica.

Desventajas web estáticas:

- ✓ Actualización de contenido debe hacerse de forma manual editando la página que almacena el servidor web.

- ✓ Limitaciones para diseño aplicaciones

Aplicaciones Web

Las aplicaciones web emplean páginas web dinámicas para crear aplicaciones que se ejecuten en un navegador.

Existen aplicaciones web para multitud de tareas.

Estas aplicaciones tienen ciertas ventajas e inconvenientes si las comparas con las aplicaciones tradicionales que se ejecutan sobre el sistema operativo de la propia máquina.

Características:

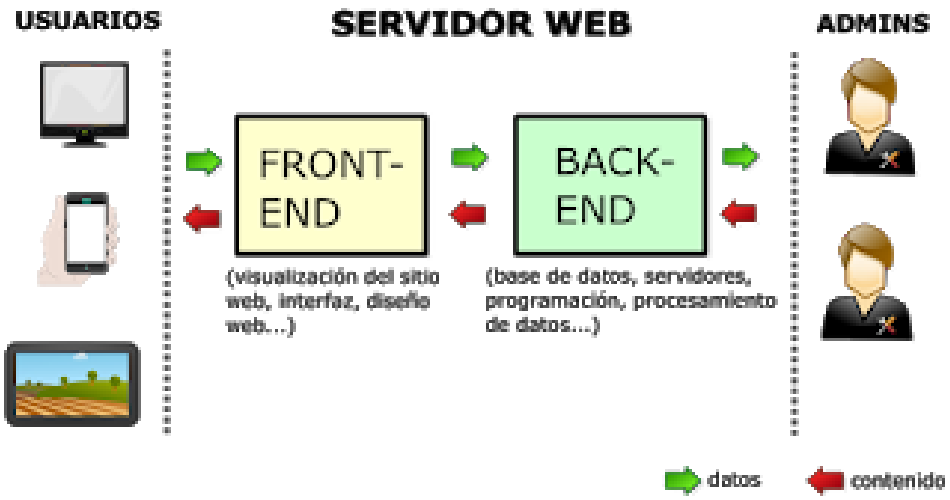
- ✓ No es necesario instalarlas en aquellos equipos en que se vayan a utilizar
- ✓ Se pueden utilizar desde cualquier dispositivo que dispongan de navegador web.
- ✓ Se pueden utilizar desde cualquier lugar en el que dispongamos de conexión con el servidor.

Aplicaciones Web

Muchas aplicaciones Web se basan en la generación de páginas dinámicas.

Esquema general:

- Una parte externa (**front-end**) conjunto de páginas que ven la gran mayoría de usuarios que las usan (usuarios externos).
- Una parte interna (**back-end)** conjunto de páginas dinámicas que utilizan las personas que producen el contenido y las que administran la aplicación web (usuarios internos) para crear contenido, organizarlo, decidir la apariencia externa, etc.



Componentes – Clientes Web (Navagadores)

Clientes Web

- ✓ Originan el tráfico web: Envían las peticiones y reciben las respuestas.
- ✓ Dos clases de clientes web: navegadores y robots
  - **Navegadores** (Iexplorer, Chrome, Opera, FireFox, Safari …).Utilizan caches de memoria y disco.
  - **Robots** (Motores de búsqueda): las peticiones son automatizadas.
- ✓ Construyen y envían la petición HTTP .
- ✓ Reciben, interpretan y presentan la respuesta.
- ✓ El protocolo por defecto es http
- ✓ Caché local: sirve recursos guardados en la caché sin conectarse al
- ✓ servidor
- ✓ Gestión de Cookies.

Componentes – Clientes Web (Navagadores)

Clientes Web

Componentes – Servidores Web

Servidores Web

Su cometido básico es proveer de contenido estático a un cliente que ha realizado una petición a través de un navegador.

Las peticiones al servidor contienen una dirección de tipo URL formada por:

- ✓ Referencia a un cierto protocolo (HTTP, FTP, etc.)
- ✓ Dirección IP o nombre de dominio del servidor
- ✓ Descripción del recurso en forma de ruta al objeto que queremos acceder.
- ✓ Opcionalmente se puede incluir el puerto por el que el servidor escucha las peticiones del cliente.

Componentes – Servidores Web

Servidores Web

Programa que contesta y genera la respuesta HTTP a las peticiones de recursos web por parte del cliente.

Funcionalidad básica:

- ✓ Se conecta con el cliente.
- ✓ Recibe el mensaje HTTP de la petición (GET, HEAD, POST)
- ✓ Procesa el mensaje HTTP .
- ✓ Localiza y envía el resultado (en forma de mensaje HTTP)

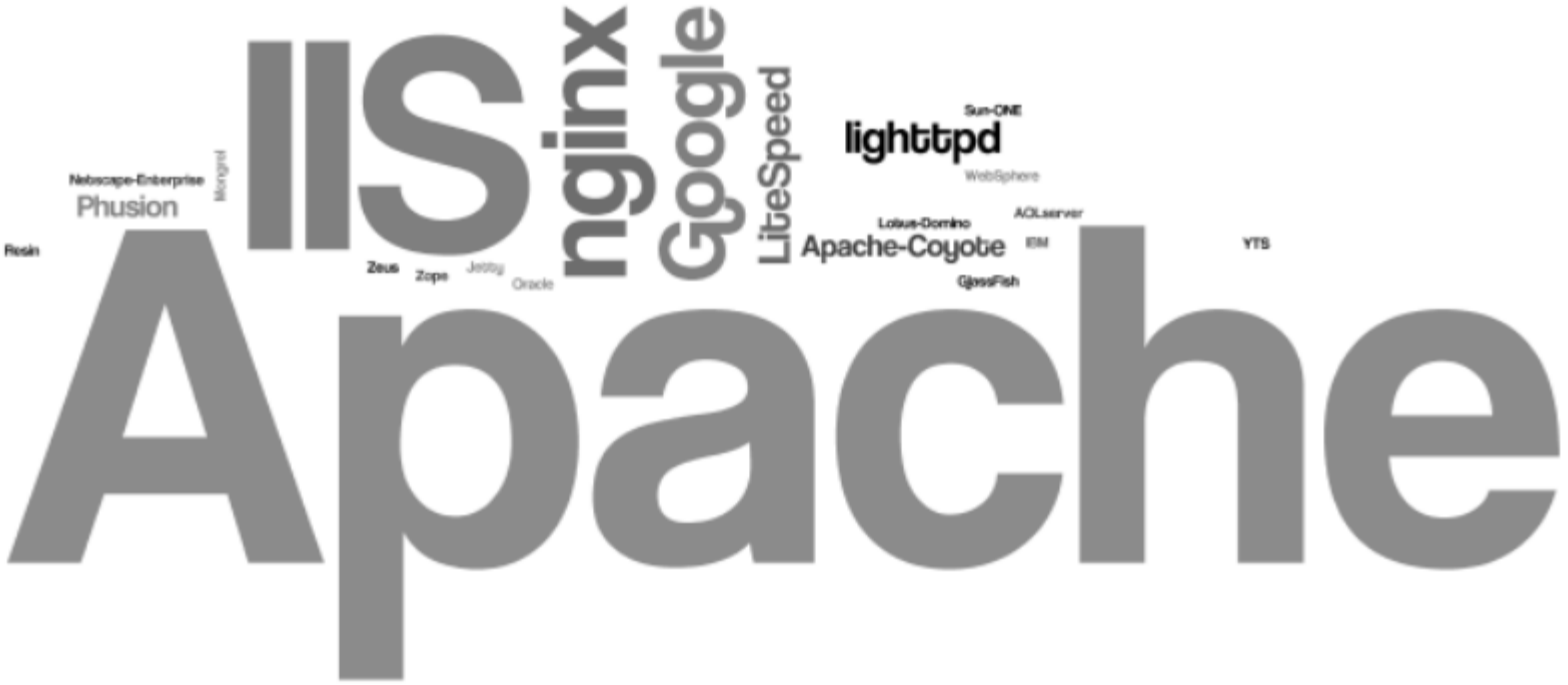
Los servidores de altas prestaciones, además:

- ✓ Tratan múltiples peticiones: hilos para manejar cada conexión.
- ✓ Generan dinámicamente contenido: ASP, PHP, JSP

Servidores Web: Apache HTTP server, IIS, Nginx, Lighttpd, …

Componentes – Servidores Web

Servidores Web



Componentes – URLs

URL (Uniform Resource Locator)

- ✓ Usado para identificar un recurso en la Web,
  Sintaxis:
  protocolo://host[:puerto]/ruta-y-nombre.

- ✓ Ejemplos

  http://www.edmodo.com
  http://192.168.204.207/ficheros.html
  http://www.daw.net:8080/datos/practica1.pdf
  ftp://ftp.rediris.es

Componentes – URLs

URI (Uniform Resource Identifier)

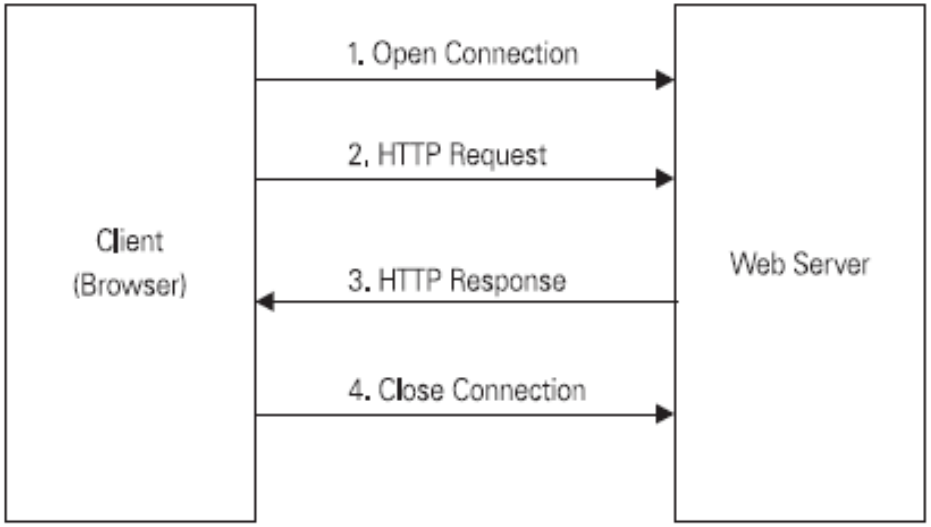
- ✓ Más general que una URL (las URLs son tipos de URIs)
- ✓ Usado para identificar un parte dentro de un recurso en la Web,
  Sintaxis:
  protocolo://host:puerto/ruta?parámetros#parte
- ✓ Ejemplos

  http://obelix.dae.es/buscarLibros.php?id=2&tema=Historia

Protocolo HTTP

- ✓ Navegador necesita interactuar con un servidor web, realiza una **request** (una solicitud o petición) hacia el servidor web, éste procesa la request y emite una **response** hacia el navegador.

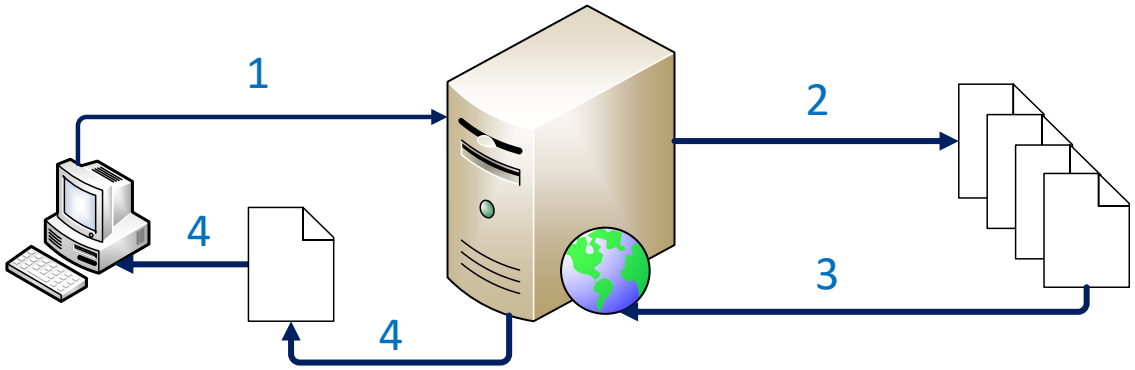
- ✓ HTTP es un **protocolo sin estado**, lo que significa que una vez realizada la request, y recibida la correspondiente response en el navegador, la conexión entre el navegador y el servidor web deja de existir (el navegador y el servidor web no permanecen conectados).



Protocolo HTTP

Peticiones HTTP

1. Desde el cliente se solicita a un servidor web un recurso (página web)
2. El servidor busca esa recurso en una ubicación
3. Si el servidor encuentra esa página, la recupera.
4. Y por último se la envía al cliente para que éste pueda mostrar su contenido.

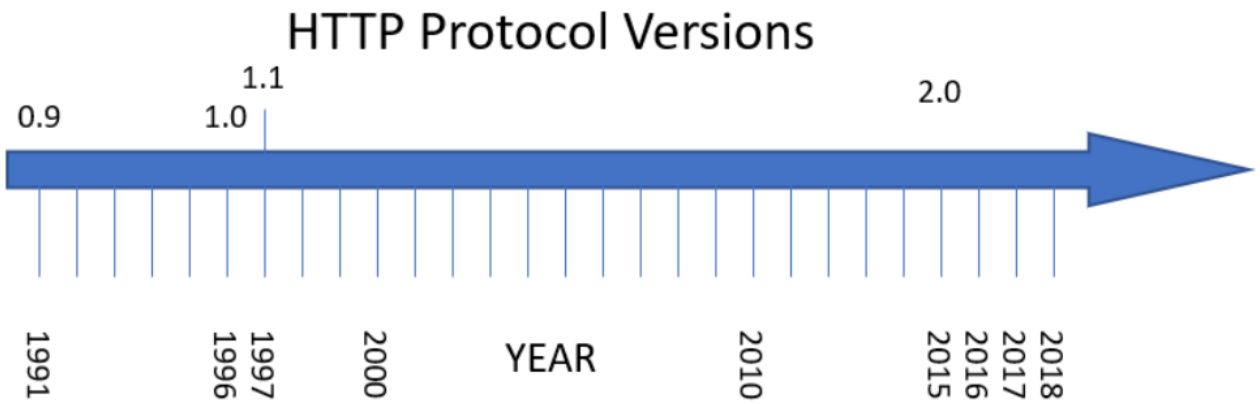


Protocolo HTTP

Versiones

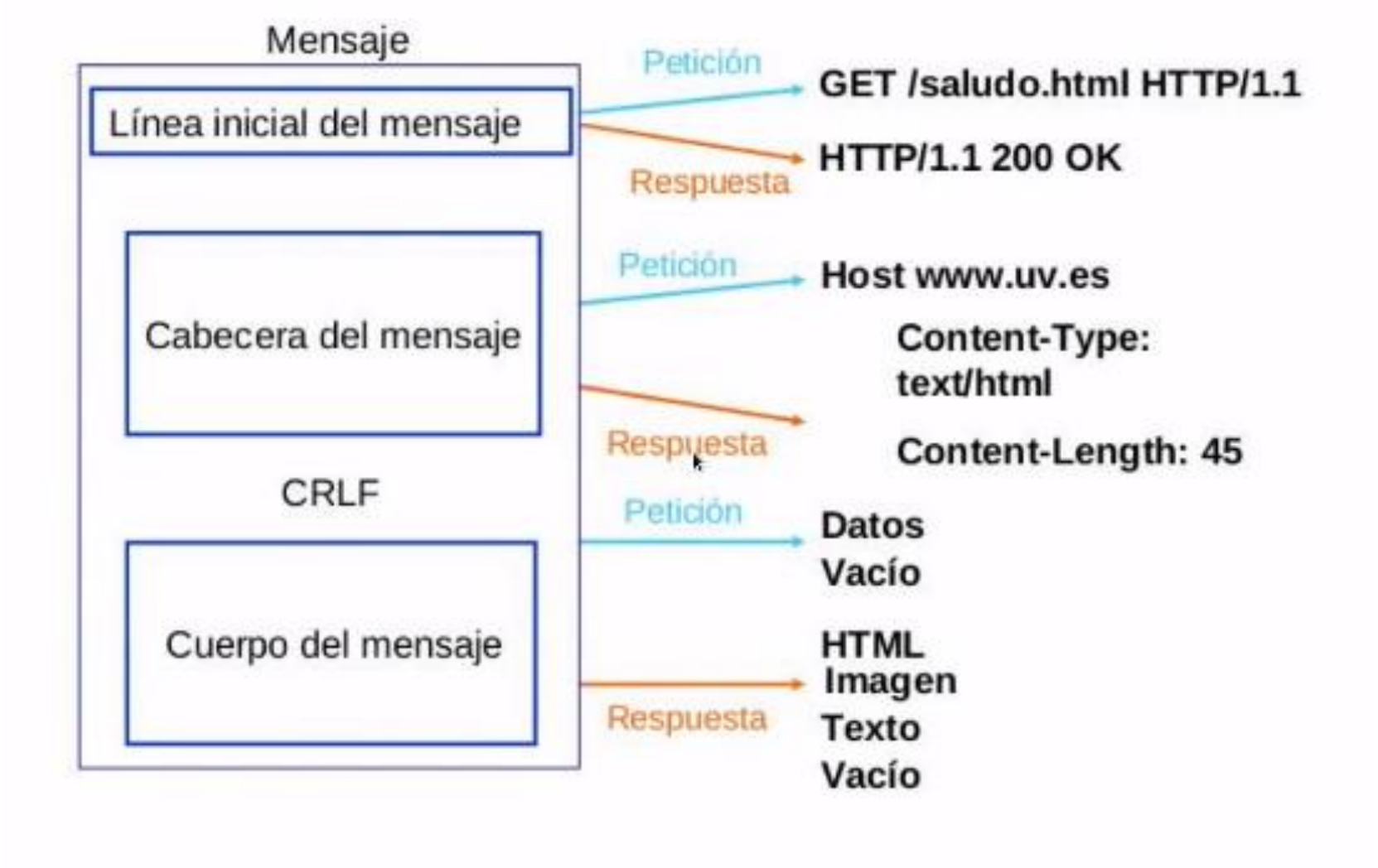
History

Year	HTTP Version
1991	0.9
1996	1.0
1997	1.1
2015	2.0



Protocolo HTTP

Estructura Mensaje



HTTP Request

Línea Principal o Inicial

- ✓ Una primera línea http que indica el método http utilizado
- ✓ URL del recurso sobre el que se quiere aplicar dicho método
- ✓ Versión HTTP utilizada por el navegador

```
GET http://www.miAplicacion.com/index.html HTTP/1.1
```

Cabecera o header: tipo navegador, idioma navegador, login de usuario, máquina del usuario….

```
Accept: image/*, application/vnd.ms-excel, */*
Accept-Language: en-ES
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: www.miAplicacion.com
Connection: Keep-Alive
```

Cuerpo (opcional): conjunto de nombres de parámetros y sus valores correspondientes

HTTP Response

Línea Principal

- ✓ Una primera línea http que indica el método http utilizado
- ✓ Código *(200 OK, 404 página no encontrada, 500 Error inesperado)*
- ✓ Descripción respuesta

```
HTTP/1.0 200 OK
```

Cabecera o header: tamaño cuerpo de la response, fecha última modificación , tipo de contenido, etc …

```
Connection: Close
Date: Fri, 02 May 2003 15:30:30 GMT
Set-Cookie: PREF=ID=1b4a0990016089fe:LD=en:TM=1051889430:
LM=1051889430:
S=JbQnlaabQb0I0KxZ; expires=Sun, 17-Jan-2038 19:14:07 GMT;
path=/; domain=.miaplicacion.com
Cache-control: private
Content-Type: text/html
Server: GWS/2.0
```

Cuerpo: recurso solicitado (página html, imagen, pdf, …)

Códigos y estados Error

- ✓ ***100 – 199:*** mensajes informativos

- ✓ ***200 – 299:*** petición procesada satisfactoriamente
  200 OK

- ✓ ***300 – 399:*** petición redirigida
  302 Found

- ✓ ***400 – 499:*** errores del cliente
  400 Bad Request: error sintaxis petición
  401 Unauthorized: usuario anónimo no autorizado
  403 Forbidden: servidor no acepta petición
  404 Not found: recurso no encontrado

- ✓ ***500 – 599:*** errores del servidor
  500 Internal Server Error: errores configuración
  502 Bad Gateway
  503 Service Unavailable: servidor no disponible

Métodos Protocolo HTTP

GET

Es el método HTTP más utilizado
Significa “quiero este recurso del servidor”
El recurso puede ser una página web estática, fichero (imagen, audio, texto)

GET http://www.miAplicacion.com/index.html HTTP/1.1

Parámetros request asociados a la petición no viajan en el cuerpo de la request sino que son añadidos a la url separados de esta por el símbolo **?**
y estos separados entre sí por **&**.

GET *http://www.miAplicacion.com/obtenerUsuarios?alta=1&edad=44* **HTTP/1.1**

De modo general:

http://servidor:puerto/aplicación/recurso?nombre1=valor1&nombre2=valor2.

Métodos Protocolo HTTP

POST

Al igual que en GET se solicita un recurso.

Al contrario que GET, los parámetros de la request no viajan en la url, sino que viajan en el cuerpo de la request

POST http://www.miAplicacion.com/index.html HTTP/1.1

Parámetros request asociados a la petición van en el cuerpo de la request (no Aparecen en la URL)

POST *http://www.miAplicacion.com/obtenerUsuarios?* **HTTP/1.1**

Métodos Protocolo HTTP

HEAD

Idéntico a GET, excepto response: servidor web no envía el cuerpo sino Header.

Este método es utilizado cuando navegador quiere saber la validez de un recurso determinado.

Esto hace que HEAD sea muy eficiente

HEAD http://www.miAplicacion.com/aplicación/pagina.html HTTP/1.1

Métodos Protocolo HTTP

PUT

Se utiliza para transmitir un recurso (un fichero) desde el cliente http hacia servidor web y ponerlo en la ruta indicada en primera línea de la request

Si el fichero ya existe en el servidor web, se sobrescribirá.

PUT **http://www.miAplicacion.com/aplicacion/curriculums/cv11.doc HTTP/1.1**

En la response de esta petición:

- Código 201 significa que el recurso no existía y fue copiado correctamente
- Código 200 significa que el recurso ha sido reemplazado

HTTP Cookies

Fragmento información envía servidor Web en Response y puede ser almacenada en cliente (si configuración lo permite)

Cliente puede enviar cookie en solicitudes posteriores al mismo servidor Web.

Cookie puede incluir:

- **Nombre (name)**: de la cookie
- **Valor (value)**: valor de la cookie
- **Fecha expiración (expires)**: fecha/hora descarte cookie
- **Ruta (path)**: dónde se almacenará la cookie

Servidores Web utilizan cookies para diferenciar usuarios/conexiones.

HTTP protocolo sin estado: navegación autenticada es posible por uso cookies

HTTP Sesiones

HTTP es un protocolo “sin estado”, por lo que cada transferencia de datos es independiente de la anterior sin ninguna relación entre ellas.

Técnicas para mantener la sesión:

- ✓ Cookies
- ✓ URL Rewriting
  http://www.daw.net/login.php?sessionid=2a1vJ
- ✓ Campos ocultos en formularios.

APIs de tecnologías: PHP, JavaEE, .NET, …

HTTP Autenticación

Mecanismos de autenticación para controlar el acceso a los recursos que ofrece el servidor.

Autenticación: aplicaciones Web deben conocer y verificar la identidad del usuario, mediante nombre de usuario y contraseña.

Conexiones persistentes: permiten que varias peticiones y respuestas sean transferidas usando la misma conexión TCp

Dos tipos:

- ✓ Basic
- ✓ Digest

Bibliografía

- ✓ Servicios de Red e Internet. Álvaro García Sánchez, Luis Enamorado Sarmiento, Javier Sanz Rodríguez. Editorial Garceta.
- ✓ http://www.w3c.org
- ✓ http://www.w3c.e