

Conceptos Básicos TCP/IP

2º DAW



2W

IES Severo Ochoa

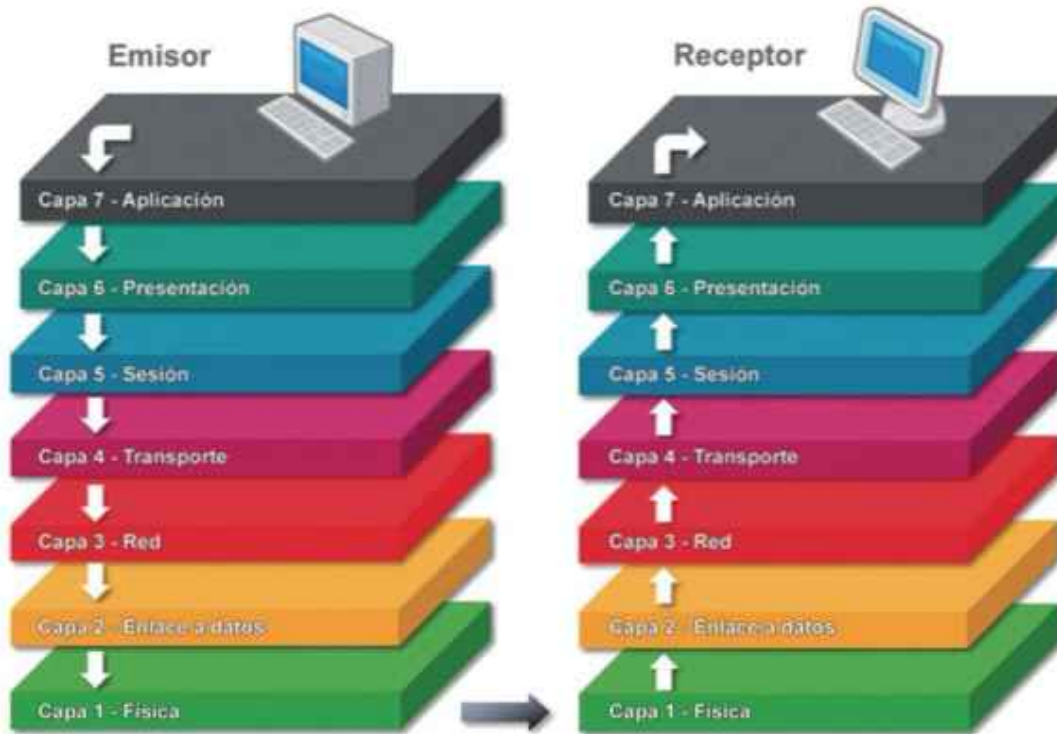
Conceptos básicos TCP/IP

- Una Red de comunicaciones esta compuesta por dos o más entidades cuya finalidad es intercambiar información.
- El comienzo fue en 1971 cuando se creó la primera red de ordenadores, conocida como **ARPANET**.
- Se crearon diferentes Redes en los años 70 sin compatibilidad entre ellas.
- Solución => Estandarizar => ISO

Modelo;Arquitectura;Protocolo

- **Modelos:** determinan la manera en la que se tiene que establecer y producir la comunicación.
- **Arquitecturas:** son el conjunto de niveles y protocolos utilizados para implementar las funciones en el proceso de comunicación.
- **Protocolos:** son el conjunto de reglas y normas que rigen las tareas para prestar un servicio.

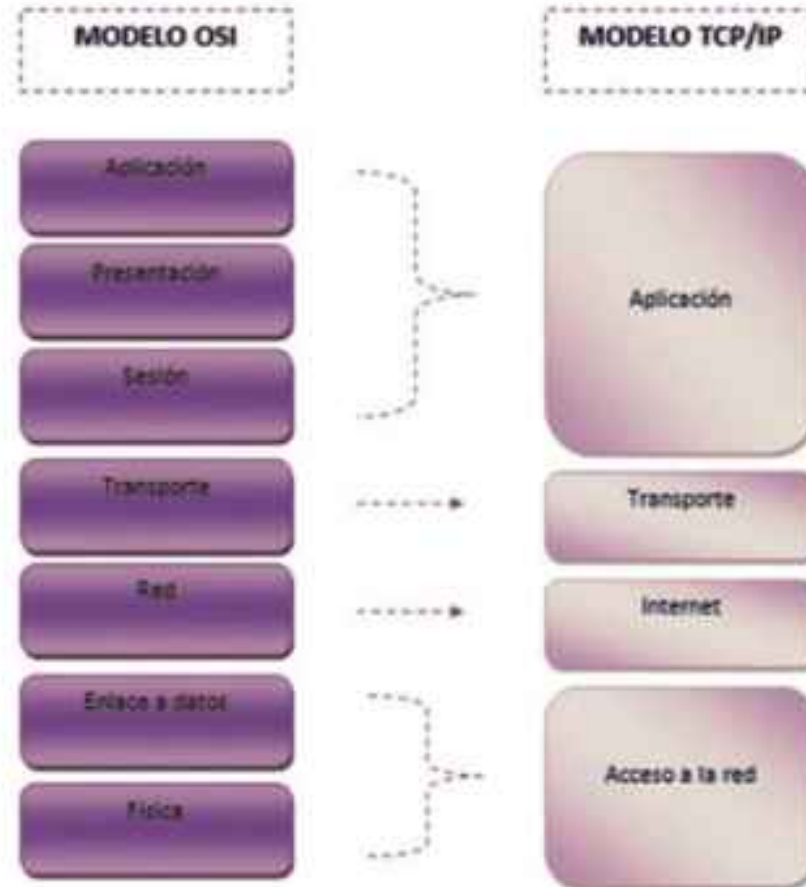
Modelo OSI



Niveles o capas



Modelo TCP/IP

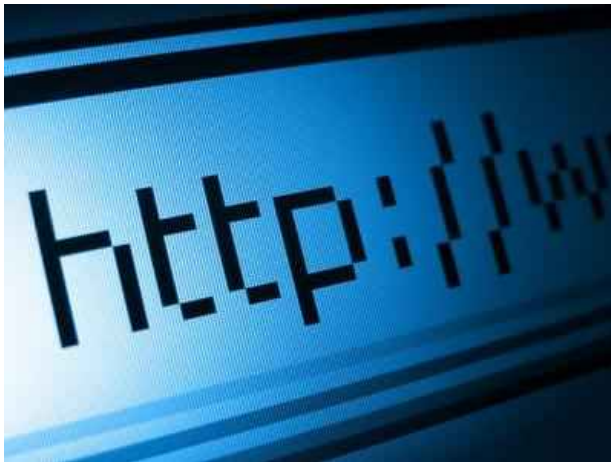


Protocolos de comunicación

CAPA	PROTOCOLO	DESCRIPCIÓN
APLICACIÓN	HTTP, HTTPS	Publicar e interpretar texto, imágenes, sonido, vídeo... en Internet.
	SMTP, POP3, IMAP	Enviar y recibir correo electrónico.
	DHCP	Configurar el equipo para obtener automáticamente una dirección IP.
	DNS	Traducir nombres de dominio a direcciones IP.
	FTP, FTPS	Transferir archivos.
	RCP	Establecer conexiones remotas para trabajar sobre otro equipo.
	SSL, TLS	Encriptar la información en el emisor para que solo el receptor pueda descifrarla.
TRANSPORTE	UDP	Enviar información de la forma más rápida posible, pero sin comprobar la llegada y sin sincronizarse.
	TCP	Enviar información estableciendo conexión previa y confirmando la llegada en el mismo orden de emisión.
INTERNET	IP	Enviar paquetes por la mejor ruta posible para que lleguen a su destino. El trazado de la ruta se realiza a través de un mecanismo llamado encaminamiento o enrutamiento.
	NAT	Traducir direcciones IP privadas en direcciones IP públicas.
ACCESO A LA RED	ETHERNET	Establecer las reglas que rigen el cableado.
	WLAN	Establecer las reglas que rigen la comunicación por WiFi.
	FDDI	Establecer las reglas para la transmisión de datos por fibra óptica en redes de área local.
	ARP, RARP	Asignar direcciones MAC con direcciones IP y viceversa.

El protocolo HTTP

- El protocolo de transferencia de hipertexto o **HTTP** (*HyperText Transfer Protocol*) establece **la sintaxis y semántica** para el intercambio de documentos **entre clientes y servidores web**.



Creado por el CERN
para **el intercambio
de información
científica de forma
rápida un coste
moderado (1990)**

El protocolo HTTP

- La primera versión referida como **HTTP/0.9** sólo permitía a la **transferencia simple de datos** (Texto) a través de Internet.
- No es hasta la versión referida como **HTTP/1.0** cuando los mensajes transferidos son enviados en formato **MIME**, Lo que implica el envío de **meta-información** conjuntamente con los datos transferidos
 - permite enviar **ficheros binarios** como texto; *imágenes, archivos multimedia, ...*



El protocolo HTTP

- Actualmente, se trata de un **protocolo estándar** que se ha convertido en la base para la **intercomunicación** de cualquier tipo de **sistemas informáticos**.
 - **Subsistemas de la empresa:** ERP / CRM
 - Pasarelas de pago.
 - Proveedores de contenidos:
 - Sistemas de publicidad
 - Venta de productos por referencia

HTTP - Funcionamiento

- El funcionamiento del **protocolo HTTP** es muy sencillo.



HTTP - Funcionamiento

- Un usuario accede a una URL introduciéndola en el **campo location** del cliente web.
 - `http://direccionIP[:puerto][path_servidor]`
 - Ejemplo: **`http://www.miweb.com/documento.html`**
- El cliente web decodifica la URL:
 - Separa las partes y realiza la **petición DNS** correspondiente.
- Abre una **conexión TCP** con el servidor y **realiza una Petición HTTP**
 - Comandos GET, POST, HEAD,...
- El **servidor devuelve la respuesta** al cliente.
 - Código de Estado
 - Tipo de Dato enviado (MIME)
- Se **cierra la conexión TCP**

Comandos HTTP

Métodos de consulta

- **GET:** Se utiliza para solicitar cualquier tipo de información/Documento al servidor
 - Ej. Hojas de estilo (**css**), documento web (**html**)
- **HEAD:** Se utiliza para obtener información sobre el recurso, **sin acceder al mismo**.
 - Ej. Tipo, fecha de modificación,... los **gestores de cache** y **servidores proxy** lo utilizan para determinar cuándo actualizar una página.



Comandos HTTP

Métodos de transferencia

- **POST:** Nos permite enviar información al servidor web con el fin:
 - Añadir contenido a un recurso existente.
 - Crear un registro nuevo en una base de datos.
 - Un ejemplo muy usual es **enviar datos recogidos de un formulario HTML.**



Ejemplo de formulario básico

file:///C:/html/formulario-basico.html

Nombre:

Edad:

Enviar datos

Abrirollave.com

Comandos HTTP

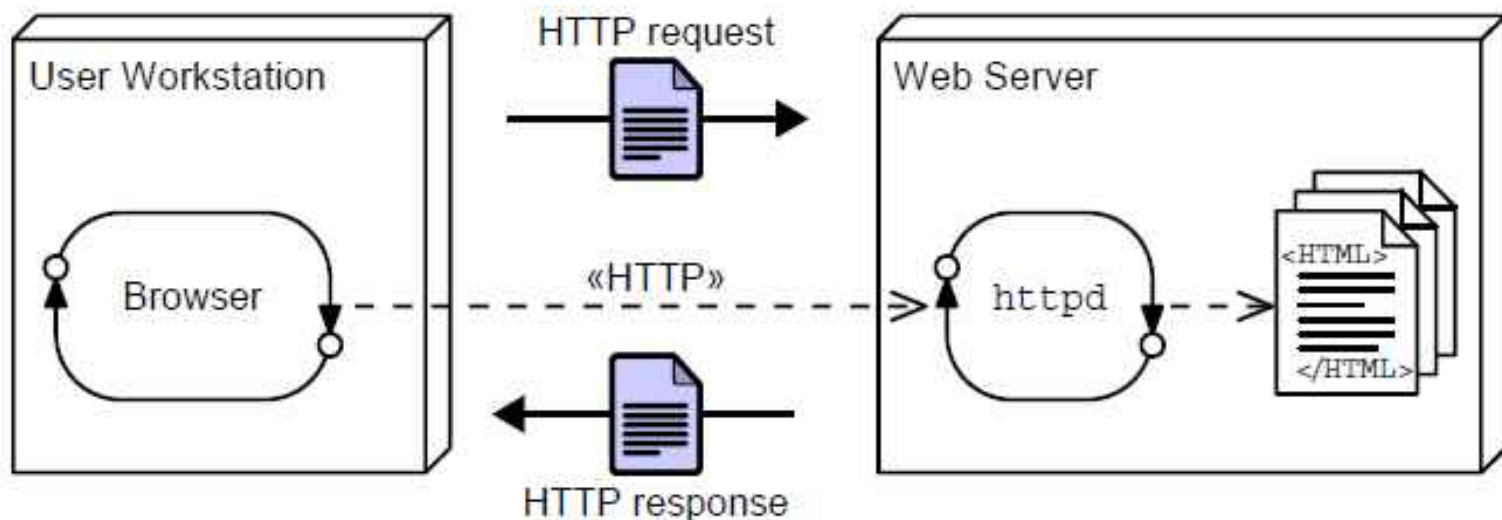
Otros métodos

- Debido al crecimiento y estandarización de el uso del **protocolo HTTP**, la **versión 1.1** incluye algunos métodos más como son:
 - **PUT**: Nos permite modificar un recurso ya existente en el servidor (modificar los datos de un usuario).
 - **DELETE**: Nos permite eliminar un recurso en el servidor.
 - **OPTIONS**: Devuelve los métodos HTTP y características que **un servidor soporta para una URL determinada**.
 - Se puede utilizar para **averiguar los recursos** de un servidor web indicando '*' en lugar de un recurso específico.

3.4 MENSAJES HTTP

Tipos

- Los **mensajes HTTP** pueden ser de dos tipos:
 - **Solicitud** (Request) del cliente al servidor.
 - **Respuesta** (Response) del servidor al cliente.



3.4.1 MENSAJES DE SOLICITUD

Estructura

- Los mensajes de solicitud son un conjunto de líneas que el cliente envía al servidor e incluyen:
 - **Línea de petición:** Especifica el **recurso solicitado**, el **método** y la **versión del protocolo**.
 - **Campos de cabecera** (Headers fields): Permiten aportar información extra a la solicitud:
 - Navegador, sistema operativo, versión del protocolo
 - **Cuerpo del mensaje:** contienen la información a transmitir (**POST** → Campos del formulario)

```
GET http://www.elpais.com HTTP/1.0
Accept: Text/html
If-Modified-Since: Saturday, 15-January-2020 14:37:11 GMT
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:23.0)
Gecko/20100101 Firefox/23.0
```

3.4.1 MENSAJES DE SOLICITUD

- Formato mensaje solicitud:

línia de petició	→	GET / HTTP/1.0	MÈTODE espai URL espai VERSIÓ CR LF
capçaleres	→	Host: www.upc.es	CAPÇALERA: espai VALOR CR LF
		...	
		User-agent: telnet	CAPÇALERA: espai VALOR CR LF
línia en blanc	→	Cr lf	CR+LF
entity body	→		En aquest cas està buit. S'utilitza amb el mètode POST (mètode usat sovint per a omplir formularis)

3.4.2 MENSAJES DE RESPUESTA

Estructura

- Los **mensajes de respuesta** son un conjunto de líneas en **modo texto** que **el servidor** envía al cliente e incluyen:
 - Una **línea de estado**: Indica **versión del protocolo, código estado/error**, texto descriptivo.
 - Campos de **cabecera** de la respuesta: Permiten aportar información extra a sobre la respuesta del servidor:
 - El cuerpo del **mensaje**: contienen la información a transmitir (**POST** → Campos del formulario)

```
HTTP/1.0 200 OK
Date: Sat, 15 Jan 2020 14:37:12 GMT
Server: MicrosoftIIS/2.0 Content-Type : text/HTML
Content-Length: 1245
Last-Modified: Fri, 14 Jan 2010 18:11:12 GMT
```

3.4.2 MENSAJE DE RESPUESTA

- Formato mensaje respuesta:

línia d'estat	→	HTTP/1.1 200 OK	VERSIÓ espai CODI_ESTAT espai FRASE cr lf
capçaleres	→	<div> <div>Date:</div> <div>...</div> <div>Content-Type:</div> </div>	CAPÇALERA: espai VALOR cr lf
línia en blanc	→	cr lf	CR+LF
entity body	→		Conté l'objecte sol·licitat

3.4.2 MENSAJE DE RESPUESTA

- Formato mensaje respuesta:

```
HTTP/1.1 200 OK
Date: Wed, 18 Sep 2019 19:44:53 GMT
Server: Apache/2.4.38 (Debian)
Last-Modified: Sat, 14 Sep 2019 17:22:37 GMT
ETag: "d-592869df67140"
Accept-Ranges: bytes
Content-Length: 13
Content-Type: text/html

<!DOCTYPE html>
<html>
<head>
<title>Mi primera web</title>
</head>
<body>
<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

Línea de Estado

Cabeceras

Cuerpo


3.4.2 MENSAJE DE RESPUESTA

- **Códigos de estado** que especifica el protocolo dependiendo de la versión utilizada:

CÓDIGOS DE ESTADO	SIGNIFICADO
100-199	Respuestas informativas
200-299	Respuestas satisfactorias
300-399	Redirecciones
400-499	Errores de los clientes
500-599	Errores del Servidor

3.4.2 MENSAJE DE RESPUESTA


- Códigos de estado “satisfactorios”:



200	OK	La solicitud se llevó a cabo de manera correcta
201	CREATED	Sigue a un comando <u>POST</u> e indica el éxito, la parte restante del cuerpo indica la dirección <u>URL</u> donde se ubicará el documento creado recientemente.
202	ACCEPTED	La solicitud ha sido aceptada, pero el procedimiento que sigue no se ha llevado a cabo
203	PARTIAL INFORMATION	Cuando se recibe este código en respuesta a un comando de <u>GET</u> indica que la respuesta no está completa.
204	NO RESPONSE	El servidor ha recibido la solicitud, pero no hay información de respuesta
205	RESET CONTENT	El servidor le indica al navegador que borre el contenido en los campos de un formulario
206	PARTIAL CONTENT	Es una respuesta a una solicitud que consiste en el encabezado <i>range</i> . El servidor debe indicar el encabezado <i>content-Range</i>

3.4.2 MENSAJE DE RESPUESTA

- Códigos de estado “redirección”:



301	MOVED	Los datos solicitados han sido transferidos a una nueva dirección
302	FOUND	Los datos solicitados se encuentran en una nueva dirección URL, pero, no obstante, pueden haber sido trasladados
303	METHOD	Significa que el cliente debe intentarlo con una nueva dirección; es preferible que intente con otro método en vez de <u>GET</u>
304	NOT MODIFIED	Si el cliente llevó a cabo un comando <u>GET</u> condicional (con la solicitud relativa a si el documento ha sido modificado desde la última vez) y el documento no ha sido modificado, este código se envía como


3.4.2 MENSAJE DE RESPUESTA

- **Códigos de estado “error del cliente”:**

400	BAD REQUEST	La sintaxis de la solicitud se encuentra formulada de manera errónea o es imposible de responder
401	UNAUTHORIZED	Los parámetros del mensaje aportan las especificaciones de formularios de autorización que se admiten. El cliente debe reformular la solicitud con los datos de autorización correctos
402	PAYMENT REQUIRED	El cliente debe reformular la solicitud con los datos de pago correctos
403	FORBIDDEN	El acceso al recurso simplemente se deniega
404	NOT FOUND	Un clásico. El servidor no halló nada en la dirección especificada. Se ha abandonado sin dejar una dirección para redireccionar... :)

3.4.2 MENSAJE DE RESPUESTA

- **Códigos de estado “error del servidor”:**



500	INTERNAL ERROR	El servidor encontró una condición inesperada que le impide seguir con la solicitud (una de esas cosas que les suceden a los servidores...)
501	NOT IMPLEMENTED	El servidor no admite el servicio solicitado (no puede saberlo todo...)
502	BAD GATEWAY	El servidor que actúa como una puerta de enlace o proxy ha recibido una respuesta no válida del servidor al que intenta acceder
503	SERVICE UNAVAILABLE	El servidor no puede responder en ese momento debido a que se encuentra congestionado (todas las líneas de comunicación se encuentran congestionadas, inténtelo de nuevo más adelante)
504	GATEWAY TIMEOUT	La respuesta del servidor ha llevado demasiado tiempo en relación al tiempo de espera que la puerta de enlace podía admitir (excedió el tiempo asignado...)

3.5 CABECERAS HTTP

- Las **cabeceras o encabezados** son campos que permiten definir la información que se va a intercambiar. Podemos clasificarlas en:
 - **Generales:** Presentes tanto en los **mensajes de petición y respuesta**.
 - **De solicitud:** Los utiliza el cliente para enviar información adicional al servidor sobre la solicitud.
 - **De respuesta:** Los utiliza el servidor para enviar información adicional al cliente.
 - **De entidad:** Proporcionan información directamente sobre la propia información que se va a transmitir.

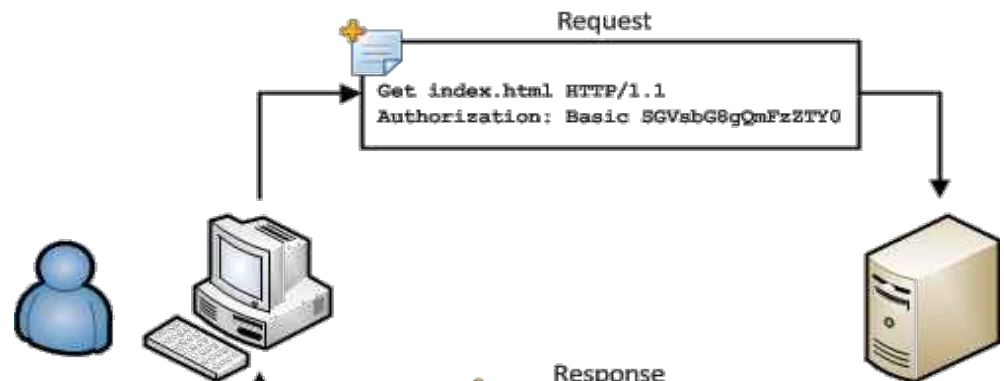
3.5 CABECERAS HTTP

- **Campos generales** (*Ejemplos*):
 - `date`: indica la hora y la fecha en que se originó el mensaje.
 - `Cache-Control`: se utiliza para especificar las directivas correspondientes a los mecanismos de caché al realizar peticiones HTTP.
 - `upgrade`: el cliente informa de las versiones del protocolo soportados, entonces el servidor elige la mejor.
 -



3.5 CABECERAS HTTP

- **Campos de petición** (*Ejemplos*)
 - **user-agent**: Este campo permite identificar la implementación del cliente http utilizado en la petición.
 - **referer**: Indica en la petición actual la dirección de la página web (URL) que ha referenciado o enlazado al recurso demandado.
 - **host**: Nombre del servidor al que va dirigida la petición, por si tiene más de uno. Permite que un servidor (una sola IP) pueda atender a diferentes nombres de dominio.

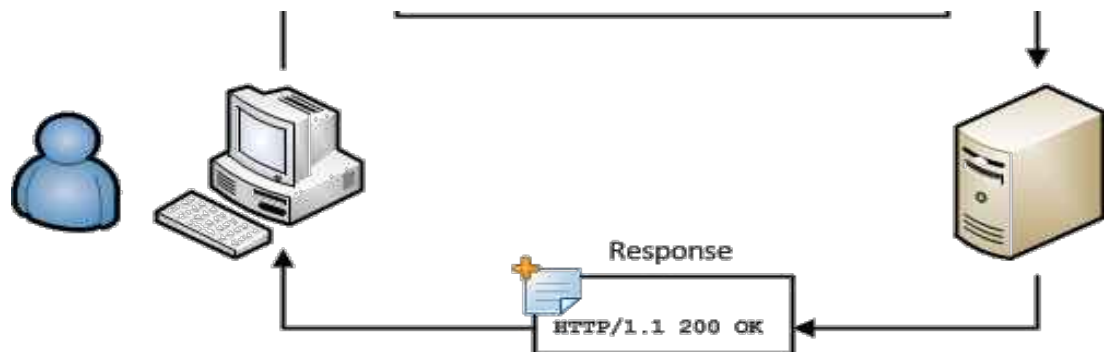


3.5 CABECERAS HTTP

- **Campos referentes a la entidad (*Ejemplos*):**
 - **Content-Length:** indica la longitud en bytes del cuerpo de la entidad.
 - **Content-Type:** Indica el tipo de contenido del cuerpo de la entidad. (*por ejemplo text/html, text/json,...*)
 - **Content-Encoding:** indica si se ha aplicado alguna codificación al cuerpo de la entidad (por ejemplo una compresión de datos **gzip**).
 - **Accept:** Tipo de contenido que acepta el cliente.
 - **Accept-Charset** (UTF-8)
 - **Accept-Encoding** (Compresion algorithm)
 - **Accept-Language** (Idiomas soportados por el cliente)

3.5 CABECERAS HTTP

- **Campos de respuesta:**
 - **Server:** similar al **user-agent**, pero referido al servidor.
 - **Location:** Indica que el recurso solicitado se debe obtener en otra dirección. Este será el caso cuando el código de respuesta sea 301 o 302.
 - **www-authenticate:** Este campo es obligatorio **cuando el código de las respuesta es 401**. Indica que es necesario presentar una credencial para acceder.



4. EXTENSIONES MIME

- **MIME** es el acrónimo inglés de **Multi-Purpose Internet Mail Extensions**, "Extensiones de correo Internet multipropósito"
- Se trata de un estándar para **adjuntar archivos** a mensajes de correo electrónico, tales como *gráficos, documentos de procesadores de texto, archivos de sonido, etc.*
 - **Transforman** los binarios a un **juego de caracteres** compatible (texto ASCII)
- Cuando se envían archivos binarios, se incluye **el tipo MIME** correspondiente en las cabeceras para cliente y servidor puedan manejar los archivos correctamente

```
Content-type: tipo_mime_principal/subtipo_mime.
```

```
Ejemplo: Content-type: image/gif
```


4. EXTENSIONES MIME

Tipos MIME primarios

- Se utilizan para clasificar la larga lista de tipos MIME existentes:
 - **Texto:** datos legibles (Ej. `text/plain`, `text/html`)
 - **Imagen:** datos binarios imagen (Ej. `image/jpeg`, `image/gif`, `image/png`)
 - **Audio:** datos binarios sonido digital (Ej. `audio/basic`, `audio/wav`)
 - **Video:** datos binarios de video (Ej. `video/mpeg`)
 - **Aplicación:** otros datos binarios (Ej. `application/octet-stream`, `application/pdf`)

4. EXTENSIONES MIME

Tipos MIME Secundarios

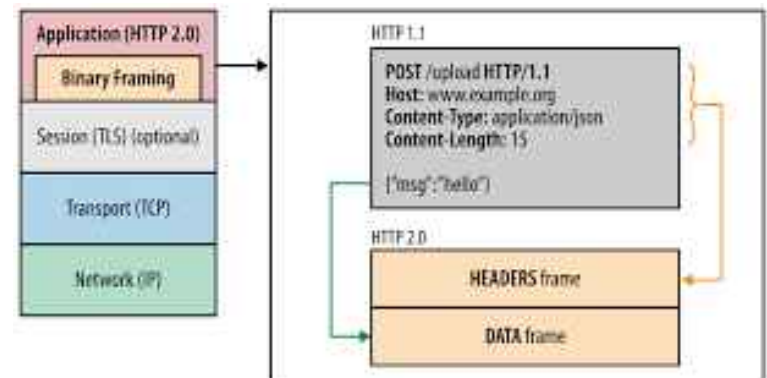
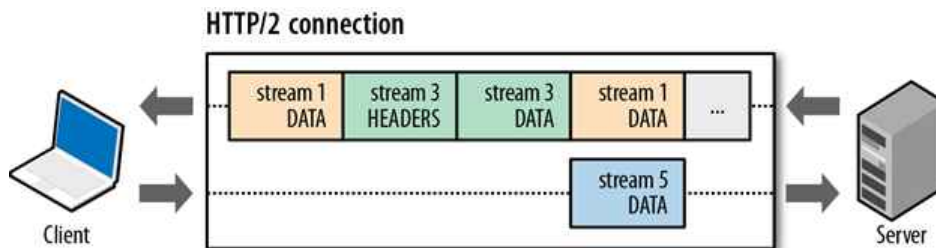
- Existe una gran lista de tipo MIME las cuales se especifican de forma detallada en los RFC 2045 al RFC 2049.

Tipo de MIME	Tipo de archivo	Extensión asociada
application/atom+xml	Archivos en formato ATOM	atom
application/iges	Archivos CAS	iges
application/javascript	Archivos JavaScript	js
application/dxf	Archivos AutoCAD	dxf
application/mp4	Archivos MPEG4	mp4
application/iges	Formato de intercambio IGES CAD	igs, iges
application/octet-stream	Archivos binarios no interpretados	bin
application/msword	Archivos de documentos Microsoft Word	doc
application/pdf	Archivos Adobe Acrobat	pdf
application/postscript	Archivos PostScript	ai, eps, ps
application/rtf	Formato de texto enriquecido	rtf
application/sgml	Archivos SGML	sgml
application/vnd.ms-excel	Archivos de hojas de cálculo Microsoft Excel	xls
application/vnd.ms-powerpoint	Archivos de presentación Microsoft Powerpoint	ppt
application/xml	Archivo XML	xml
application/x-tar	Archivos TAR comprimidos	tar
application/zip	Archivos ZIP comprimidos	man

audio/basic	Archivos de audio básicos	au, snd
audio/mpeg	Archivo de audio MPEG	mpg, mp3
audio/mp4	Archivo de audio MPEG-4	mp4
audio/x-aiff	Archivos de audio AIFF	aif, aiff, aifc
audio/x-wav	Archivos de audio Wav	wav
image/gif	Imágenes Gif	man
image/jpeg	?Imágenes Jpeg	jpg, jpeg, jpe
imagen/png	Imágenes PNG	png
image/tiff	?Imágenes Tiff	tiff, tif
image/x-portable-bitmap	Archivos Bitmap PBM	pbm
image/x-portable-graymap	Archivos Graymap PBM	pgm
image/x-portable-pixmap	Archivos Pixmap PBM	ppm
multipart/x-zip	Archivos comprimidos en Zip	zip
multipart/x-gzip	Archivos comprimidos en Zip GNU	gz, gzip
text/css	Hoja de estilo	css
text/csv	Archivos de texto separados por comas	csv
text/html	Archivos HTML	htm, html
text/plain	Archivos de texto sin formato	txt, g, h, c, cc, hh, m, f90

5. HTTP 2.0

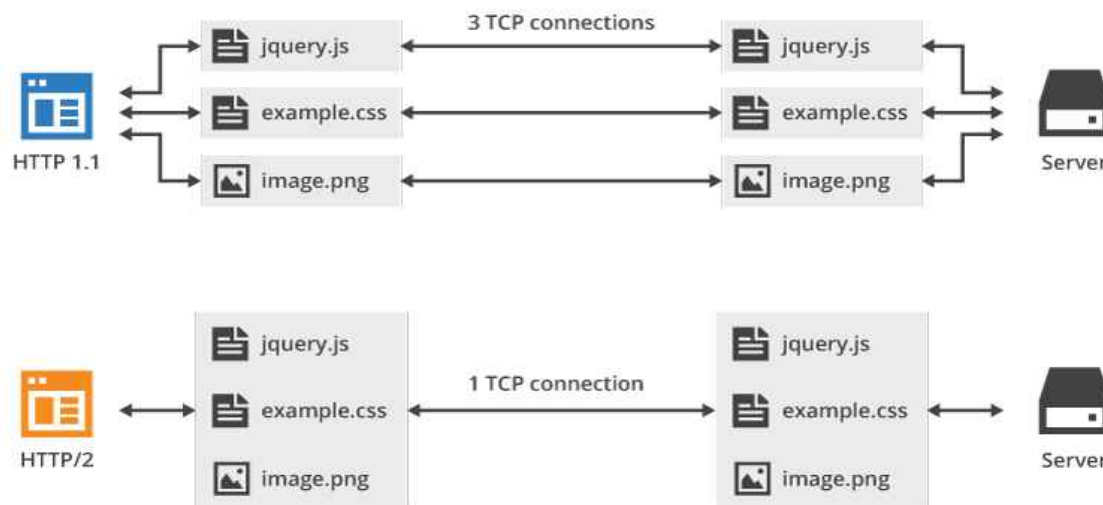
- Desarrollado por **Google** con la intención de **reducir el tiempo de carga** de páginas utilizando diferentes técnicas
- Publicación del **estándar** en **2015**
- Se añade una **capa a nivel de aplicación** que encapsula todos los mensajes en **formato binario** manteniendo toda la semántica anterior.
- Métodos / Comandos
- Cabeceras



5. HTTP 2.0

Características

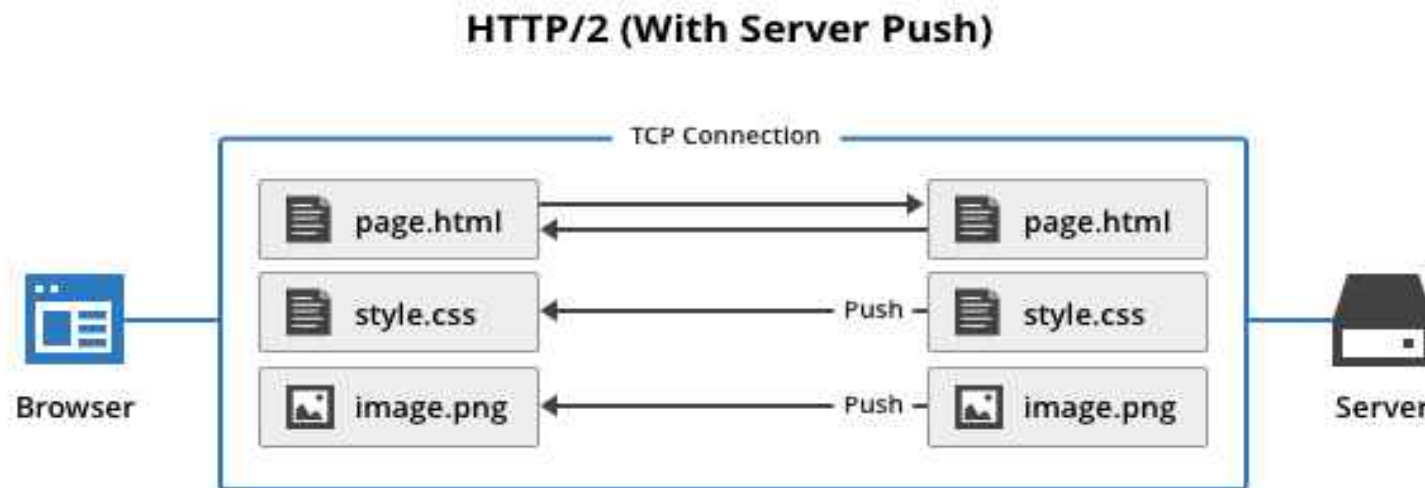
- **Formato binario:** en lugar de texto.
- **Compresión:** las cabeceras son comprimidas y empaquetados en un único bloque comprimido que es enviado como una unidad.
- **Multiplexación:** permite enviar diferentes peticiones HTTPs sobre la misma conexión TCP.



5. HTTP 2.0

Características.

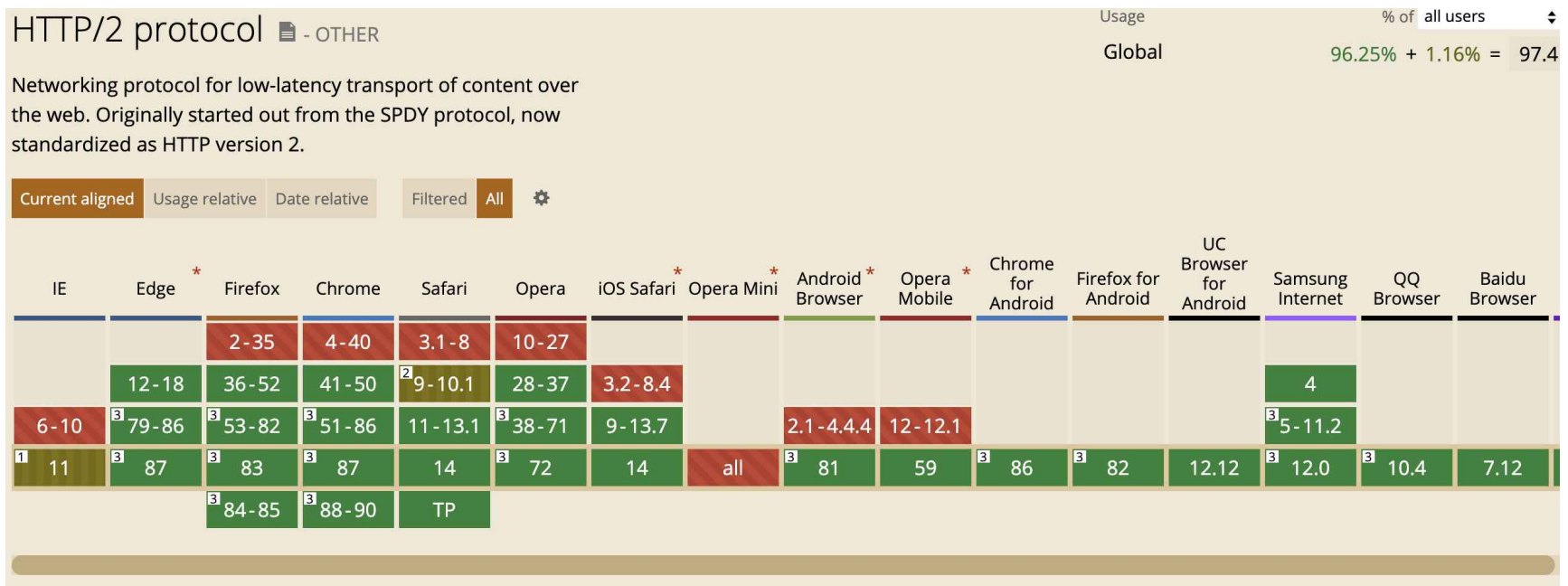
- **ServerPush:** permite enviar recursos a la cache del navegador sin que este los solicite reutilizando la conexión con el cliente, enviándole recursos que probablemente vaya a necesitar.



Single TCP Connection, Single HTTP Request

5. HTTP 2.0

- Actualmente está **soportado** por la mayoría de los navegadores, pero solo cuando utilizan **SSL/TLS** (HTTPS).



¿Alguna pregunta?