TEMA 5: La capa de aplicación I: World Wide Web.

5.1 Introducción.

Inicialmente la idea del World Wide Web[1] surgió en el laboratorio de altas energías del CERN, el Centro Europeo de Investigación Nuclear. La mayoría de los experimentos, altamente complejos y que requieren años de planteamiento y construcción de equipo involucran a equipos multidisciplinares formados por personas de distintos países europeos. La Web surgió por la necesidad de lograr que estos equipos de investigadores, dispersos geográficamente por distintos países, tuvieran la posibilidad de colaborar de forma rápida y eficaz en el diseño y desarrollo de un conjunto rápidamente cambiante de informes, planos, dibujos, fotos y otros documentos.

La propuesta inicial de la Web de documentos vinculados surgió del físico del CERN Tim Berners-Lee en marzo de 1989. El primer prototipo (basado en texto) estaba en funcionamiento 18 meses después. En diciembre de 1991 se hizo una demostración pública en la conferencia Hypertext'91 en San Antonio, Texas. El desarrollo continuó durante el siguiente año, culminando con la liberación de la primera interfaz gráfica, Mosaic, en febrero de 1993.

Mosaic tuvo tanto éxito que, un año después, su autor, Marc Andreeseen, dejó el Centro Nacional de Aplicaciones de Supercómputo, donde se desarrolló Mosaic, para formar una compañía, Netscape Communications Corp., cuya meta fue desarrollar clientes, servidores y otros tipos de software de la Web. Cuando Netscape se volvió pública en 1995, los inversionistas, pensando que esta era la siguiente Microsoft, pagaron 1.500 millones de dólares por las acciones. Esta marca fue aún más sorprendente porque la compañía sólo tenía un producto, operaba con grandes pérdidas y había anunciado en su prospecto que no esperaba tener ganancias en un futuro cercano.

En 1994, el CERN y el M.I.T. firmaron un acuerdo para establecer el World Wide Web Consortium, una organización dedicada al desarrollo de la Web, la estandarización de protocolos y el fomento de interoperabilidad entre las instalaciones. Tim Berners-Lee se convirtió en el director.

Actualmente el Web es la herramienta más conocida y utilizada en la red Internet, siendo además la que más ha contribuido a popularizar la misma y fomentar su uso.

5.2 El protocolo HTTP.

El HTTP (HyperText Transfer Protocolo) es la base del armazón arquitectónico que se conoce actualmente como el Web. El servicio HTTP esta basado en una arquitectura cliente/servidor. En esta arquitectura la mayor complejidad, tanto de desarrollo como de administración, radica en el servidor, siendo el cliente tan solo una

[1] A partir de ahora lo denominaremos simplemente como Web.

herramienta capaz de ofrecer por pantalla los distintos elementos (documentos de texto, archivos de imágenes, etc.), que envía el servidor al cliente.

Cada interacción HTTP consiste en una solicitud ASCII seguida de una respuesta de tipo MIME RFC 822[2]. Aunque la conexión de transporte se realiza mediante el protocolo TCP, el estándar no requiere formalmente su uso.[3]

El protocolo HTTP consiste en dos elementos bastante diferentes: las solicitudes de los clientes a los servidores y las respuestas en el otro sentido. Aunque el HTTP se desarrollo inicialmente para usarse en la Web, ha sido generalizado con posterioridad en previsión de su utilización en futuras aplicaciones orientadas a objetos. Por esta razón, la primera palabra de la línea de solicitud completa es sencillamente el nombre del método (comando) a ejecutar con la página de la Web (u objeto general). Los métodos existentes se listan en la tabla siguiente, siendo sensibles al contexto (mayúsculas y minúsculas), por lo cual *GET* es un método valido pero *get* no lo es.

Método	**Descripción**
OPTIONS	Solicita información sobre las opciones de comunicación.
GET	Solicita recibir una página Web.
HEAD	Solicita leer la cabecera de una página Web.
POST	Añade información a un recurso nombrado.
PUT	Solicita almacenar una página Web.
DELETE	Elimina una página Web.
TRACE	Invoca la devolución del mensaje de solicitud.

El método *OPTIONS* solicita al servidor información sobre las opciones de comunicación disponibles para el recurso apuntado por un URL, generalmente un tipo MIME (text/html, etc.). De esta forma, el cliente puede determinar las posibilidades que tiene el servidor o las opciones asociadas a un recurso determinado.

El método *GET*[4] solicita al servidor que envíe la página codificada adecuadamente en MIME. Sin embargo, si a la solicitud *GET* le sigue una cabecera *If-Modified-Since*, el servidor sólo envía los datos si fueron modificados después de la fecha proporcionada. Usando este mecanismo, un visualizador al que se solicitó una página que está en caché puede realizar una solicitud condicional al servidor.

El método *HEAD* simplemente pide la cabecera del mensaje, sin la página. Este método puede servir para obtener la hora de la última modificación, para recolectar información con fines de indexación, o simplemente para comprobar la validez de un URL.

El método *POST* se utiliza para solicitar al servidor que acepte la información que se envía adjunta al mensaje. Este método se utiliza generalmente para la publicación de un mensaje en un grupo de noticias y para proporcionar un bloque de datos al servidor (por ejemplo los datos rellenados en un formulario por el usuario).

[2] El RFC 822 describe el formato estándar de intercambio de correo.
[3] Actualmente todos los servidores de HTTP utilizan el protocolo de transporte TCP, por lo que el desarrollo de un servidor o cliente que no utilice dicho protocolo de transporte no es aconsejable.
[4] Con posterioridad veremos más detalladamente el método GET.

El método *PUT* es el inverso de *GET*, en lugar de leer una página la escribe. Este método hace posible construir un conjunto de páginas de la Web en un servidor remoto. El cuerpo de la solicitud contiene la página y puede codificarse usando MIME, en cuyo caso las líneas que siguen a *PUT* deben incluir cabeceras *Content-Type* y de validación de identificación, para demostrar que el solicitante tiene permisos de ejecución de la operación.

El método *DELETE* elimina la página. Como con *PUT*, la validación de identificación y los permisos desempeñan un papel principal. No hay garantía de que *DELETE* tendrá éxito, puesto que, incluso si el servidor HTTP remoto está dispuesto a borrar la página, el archivo subyacente puede tener un modo que prohíba al servidor HTTP su modificación o eliminación.

Por último, el método *TRACE* se utiliza para depurar aplicaciones. El servidor final debe devolver el mensaje de solicitud, reflejando que ha recibido de forma correcta el mensaje o bien el tipo de error detectado.

Cada solicitud recibe una respuesta que consiste en una línea de estado y, posiblemente, información adicional (por ejemplo, toda o parte de una página Web). La línea de estado contiene un código que consiste en un número de tres dígitos y, posiblemente, un mensaje de texto aclarativo del significado del código numérico. Un ejemplo de línea de estado es el siguiente:

HTTP/1.0 200 OK

Existen cinco tipos de códigos en función del primer dígito:

Código	**Descripción**
1xx	Informativo. No utilizado, reservado para usos futuros.
2xx	Éxito. La acción fue recibida y aceptada.
3xx	Redirección. Se necesita una acción adicional para llevar a cabo la solicitud.
4xx	Error del cliente. La solicitud contiene sintaxis errónea o no se puede conceder.
5xx	Error del servidor. El servidor no puede atender una solicitud aparentemente correcta.

El HTTP evoluciona constantemente. Se usan varias versiones y se están desarrollando otras. Las versiones se especifican mediante un sistema de numeración del tipo *<mayor>.<menor>* para indicar las versiones del protocolo. De esta forma el emisor puede indicar el formato del mensaje y su capacidad para entender futuras comunicaciones HTTP. La versión del mensaje HTTP se indica en el campo *HTTP-Version* en la primera línea del mensaje, como en el siguiente ejemplo:

HTTP-Version: HTTP/1.0

En caso de no especificarse la versión del protocolo, el receptor del mensaje asume que el mensaje tiene el formato *HTTP/1.0*.

Las dos versiones principales existentes actualmente son la *HTTP/1.0* y la *HTTP/1.1*. La diferencia principal entre ambas es que, mientras la versión 1.0 obliga a que cada petición que un cliente realiza a un servidor genere una conexión TCP

diferente, la versión 1.1 permite que una conexión albergue diferentes intercambios de solicitudes y respuestas.

5.2.1 El método GET.

De todos los métodos explicados con anterioridad, el método más usado es el método *GET*, que como hemos visto permite la solicitud de una página Web a un servidor por parte de un cliente. Las versiones actuales de HTTP reconocen dos tipos de solicitudes distintas del método *GET*: solicitudes sencillas y solicitudes completas.

Las solicitudes sencillas consisten en una única línea que comienza con el método *GET* y a continuación se encuentra el nombre de la página deseada, sin especificar la versión del protocolo y sin ningún dato adicional. Por tanto, su sintaxis es:

GET <página solicitada>

La respuesta que se obtiene del servidor no incluye ninguna línea con el estado, esto es, con el código de acierto o error de la solicitud enviada, consistiendo simplemente en una página sin ningún tipo de cabecera, sin ningún formato MIME y sin codificación alguna. Un ejemplo de solicitud sencilla es:

GET /home.html

Obteniendo como respuesta las siguientes líneas, que como puede observarse no van precedidas de ninguna línea de estado:

<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
…
</body>
</html>

Las solicitudes completas, las más usadas en la actualidad, se indican por la presencia de la versión del protocolo en la línea del método *GET*. A continuación viene una línea que indica el nombre del ordenador[5] al que se le realizó la petición de la página, y un conjunto de líneas subsiguientes que informan sobre la versión del cliente Web que envió la solicitud, los formatos MIME que son aceptados en la respuesta, etc. Por tanto, la sintaxis de una solicitud completa es:

GET <página solicitada> <versión del protocolo HTTP>
Host: <nombre del servidor Web>
...

Dos ejemplos de solicitudes completas de páginas Web son los siguientes:

[5] La utilidad de que en dicha línea figure el nombre del ordenador y no la dirección IP se entiende dentro del contexto de los dominios virtuales, esto es, un mismo ordenador que posee las páginas Web de varios dominios (nombre) Web distintos.

GET /home.html HTTP/1.1
Host: robotica.uv.es:
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.1) Gecko/20021003
Accept:
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,video/x-mng,image/png,image/jpeg,image/gif;q=0.2,text/css,*/*;q=0.1
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip, deflate, compress;q=0.9
*Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66*
Keep-Alive: 300
Connection: keep-alive

GET /prueba.html HTTP/1.1
Host: www.cdlibre.org
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.1) Gecko/20021003
Accept:
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,video/x-mng,image/png,image/jpeg,image/gif;q=0.2,text/css,/*;q=0.1*
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip, deflate, compress;q=0.9
*Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66*
Keep-Alive: 300
Connection: keep-alive

La respuesta recibida contiene información sobre si la solicitud se proceso de forma correcta, la fecha y hora de la solicitud, fecha y hora de la última modificación de la página Web solicitada, tamaño de la página, formato MIME de la página enviada, etc. En nuestro caso, la respuesta a una de las solicitudes anteriores es la siguiente[6]:

HTTP/1.1 200 OK
Date: Tue, 01 Apr 2003 11:44:16 GMT
Server: Apache/2.0.40 (Red Hat Linux)
Last-Modified: Mon, 23 Dec 2002 18:24:56 GMT
ETag: "16ba5-184c-60452e00"
Accept-Ranges: bytes
Content-Length: 6220
Connection: close
Content-Type: text/html; charset=ISO-8859-1
...

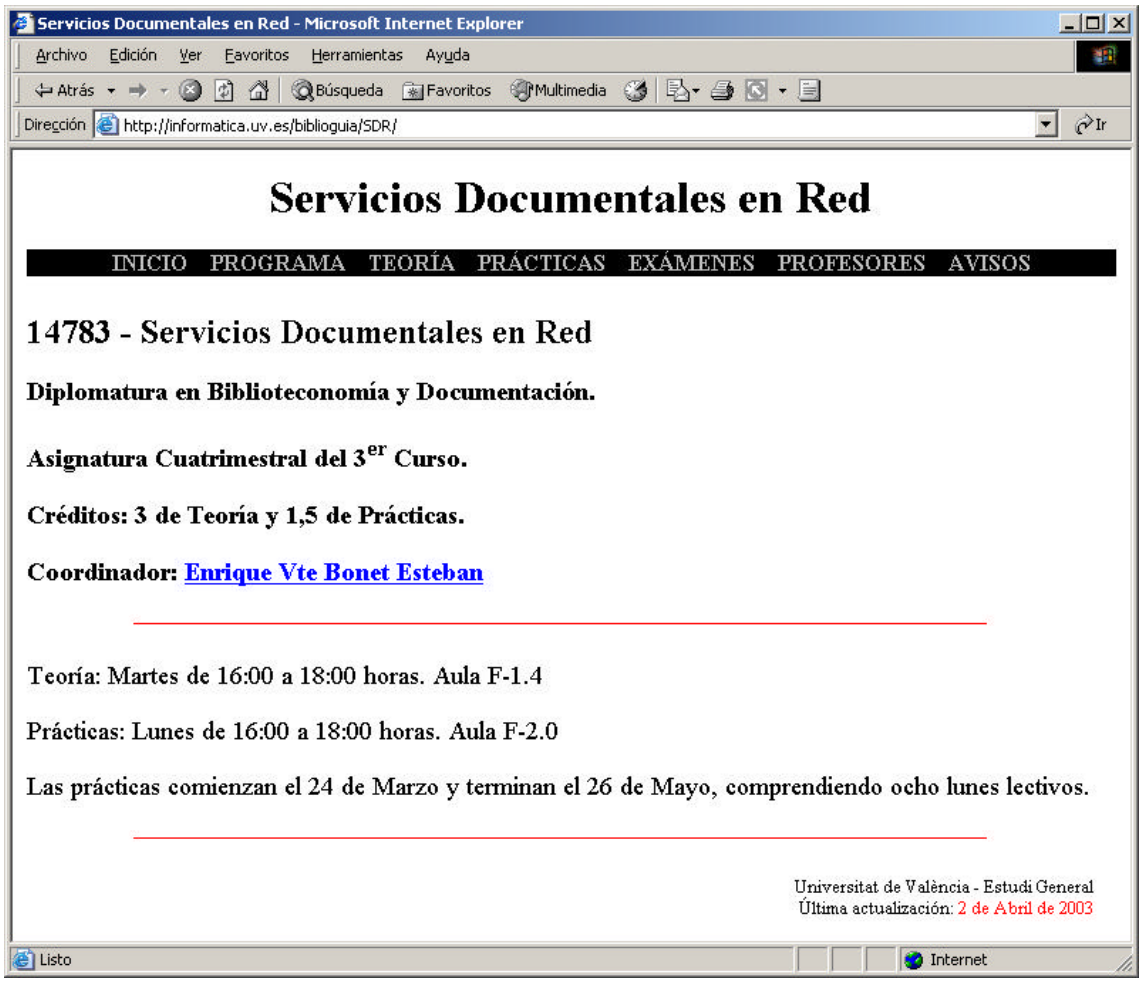
5.3 El cliente Web.

Desde el punto de vista del usuario, la Web consiste en un enorme conjunto a nivel mundial de documentos, llamados páginas. Cada página puede contener vínculos (enlaces) con otras páginas situadas en cualquier lugar del mundo. Los usuarios pueden

[6] Seguidas dichas líneas de cabecera, obviamente, del contenido de la página HTML solicitada, que aquí no es mostrada.

seguir un vínculo (por ejemplo, haciendo clic en él), lo que los lleva a la página vinculada. Este proceso puede repetirse indefinidamente.

Las páginas Web se ven mediante un programa llamado navegador[7]. El navegador solicita una página Web, espera la recepción de la página solicitada y, una vez obtenida, interpreta el texto recibido[8] y los comandos para dar formato al texto que contiene la página y la muestra en la pantalla. Un ejemplo de página Web puede verse a continuación:



En la actualidad, existen páginas que contienen pistas de audio, fragmentos de vídeo, etc[9]. En tal caso los navegadores revisan un archivo de configuración para ver el modo de mostrar dichos datos al usuario. Generalmente el archivo de configuración indica el nombre del programa, llamado visor externo o aplicación ayudante, que se ejecutará con la página Web traída como entrada. Si no existe un visor para ese tipo de datos el navegador solicita al usuario que escoja uno.

La configuración de un cliente Web es relativamente sencilla, bastando con instalar de forma correcta el navegador, así como todos los visores externos que

[7] Actualmente los navegadores más utilizados son el Internet Explorer (Windows) y el Mozilla (Linux).
[8] Inicialmente las páginas Web estaban escritas mediante HTML. En la actualidad las páginas Web han evolucionado mezclado otros lenguajes como Javascript, etc.
[9] El resultado de mezclar páginas de hipertexto con otros medios se conoce con el nombre de hipermedia.

queramos utilizar, para que este funcione. La única dificultad que puede presentarse es la necesidad de configurar el servidor proxy[10] de la red, de forma que se nos permita el acceso a las páginas Web externas a nuestra subred[11].

Dicha configuración es sencilla, así en el servidor Web Microsoft Explorer la configuración se realiza mediante el acceso a los siguientes menús y submenús: Herramientas → Opciones de Internet → Conexiones → Configuración de LAN. Una vez aquí, marcamos la casilla "Utilizar un servidor proxy para su LAN..." , y si deseamos también la casilla "No utilizar servidor proxy para direcciones locales", pulsamos en Opciones Avanzadas y en la línea de HTTP ponemos como servidor proxy proxy.uv.es y como puerto el 8080.

De igual forma, en el servidor Web Mozilla, distribuido generalmente con el sistema operativo Linux, la configuración se realiza mediante el acceso a los menús y submenús: Edición → Preferencias → Avanzadas → Proxys → HTTP Proxys. Una vez aquí marcamos la casilla "Configuración Manual del Proxy" y escribimos en la línea de Proxy HTTP el servidor proxy.uv.es con puerto 8080. Además, podemos poner en la línea de "No Proxy para:" la red uv.es, pues corresponde a nuestra red local.[12]

La solicitud de una página Web determinada se realiza mediante la introducción en el navegador Web de lo que se conoce como "Localizador Universal de Recursos", URL (Universal Resource Locator). El URL de una página Web esta formado por tres campos, de acuerdo a la siguiente sintaxis:

http://<nombre del ordenador>[:puerto][/<página Web solicitada>]

Donde *"nombre del ordenador"* es el nombre del ordenador donde se encuentra la página Web que deseamos visualizar. *"puerto"* es un parámetro optativo que indica el número de puerto TCP que utiliza el servidor de HTTP[13] y *"página Web solicitada"* es el nombre de la página Web solicitada. Este parámetro es optativo, aunque suele aparecer casi siempre, pues en ciertas configuraciones del servidor si no es solicitada ninguna página Web, el servidor asume por defecto que se solicita una página Web de una lista de páginas Web por defecto que tiene configurada, devolviendo la primera de las páginas Web por defecto que se encuentren según el orden preestablecido.

5.3 El servidor Web.

Cada instalación de la Web tiene un proceso servidor que escucha, de forma general, el puerto TCP 80, esperando conexiones entrantes de los clientes (normalmente

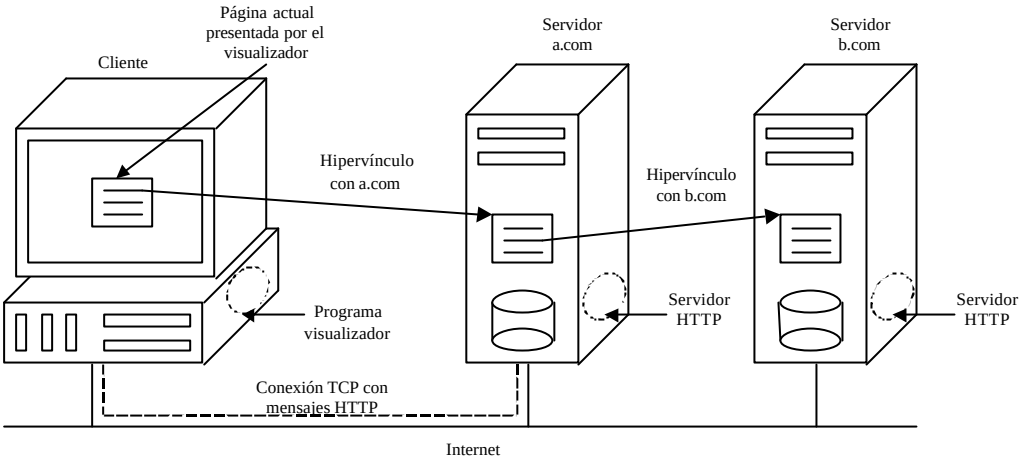
[10] Un servidor proxy es un programa que gestiona las conexiones Web de una red, almacenando las páginas recibidas, de forma que posteriores peticiones de las mismas páginas Web no tengan que ser solicitadas al servidor que las contiene, sino que el programa proxy proporcione dichas páginas Web, mejorando la velocidad de respuesta y disminuyendo la congestión en la red.

[11] Tal y como sucede en la Universitat de València, en la cual el acceso externo al servicio Web debe realizarse a través de un servidor proxy situado en el puerto TCP 8080 del ordenador proxy.uv.es.

[12] En ambos casos hemos supuesto que deseamos hacer la configuración de forma manual. Existen otras opciones que permiten detectar de forma automática el servidor proxy de la red, y para las cuales basta con activar la casilla adecuada.

[13] El puerto 80 TCP es el que tiene asignado por defecto el servicio de Web. Sin embargo, un servidor Web puede instalarse en cualquier puerto TCP que se encuentre disponible.

navegadores Web). Tras establecerse una conexión, el cliente envía una solicitud y el servidor envía una respuesta, liberándose a continuación la conexión. Un ejemplo sencillo de uso puede proporcionar una idea razonable sobre el funcionamiento de los servidores de la Web. Veamos la figura siguiente:



Supongamos que el usuario acaba de hacer clic en alguna parte del texto o en un icono que apunta a una página cuyo nombre URL (Uniform Resource Locator) sea *http://www.uv.es/index.html.* Los pasos que se ejecutan entre el clic del usuario y la presentación de la página son los siguientes:

1. El visualizador determina el URL.
2. El visualizador solicita al DNS la dirección IP de *www.uv.es*
3. El DNS contesta con 147.156.1.46.
4. El visualizador establece una conexión TCP con el puerto 80 de 147.156.1.46.
5. A continuación, el visualizador emite un comando *GET /index.html HTTP/1.0.*
6. El servidor *www.uv.es* envía el archivo *index.html.*
7. Se libera la conexión TCP.
8. El visualizador presenta todo el texto de *index.html.*
9. El visualizador trae y presenta todas las imágenes de *index.html.*

Muchos visualizadores presentan el paso que están ejecutando en cada momento en una línea de estado en la parte inferior de la pantalla, de forma que el usuario puede ver que esta haciendo y si sucede algún error a que es debido. Es necesario resaltar que para cada imagen en línea (icono, dibujo, fotografía, etc.) de una página, el visualizador establece una conexión TCP nueva con el servidor, con lo cual si una página contiene muchos iconos, todos en el mismo servidor, el establecer, usar y liberar una conexión nueva para cada uno no es muy eficiente, pero simplifica la implementación.

Puesto que HTTP es un protocolo ASCII como el SMTP, es bastante fácil que una persona en una terminal hable directamente con servidores de la Web. Todo lo que se necesita es una conexión TCP al puerto 80 del servidor. La manera más sencilla es usar el programa telnet, del cual mostramos un ejemplo a continuación. El ejemplo muestra como un cliente, en este caso una persona, pero generalmente un visualizador, se conecta con el ordenador y le envía un comando solicitando una página en particular. Las líneas marcadas con C: son tecleadas por el usuario (cliente), las líneas marcadas

con T: son producidas por el programa telnet usado para la comunicación y las líneas marcadas con S: son producidas por el servidor Web.

C: telnet www.uv.es 80
T: Triying 147.156.1.46…
T: Connected to www.uv.es.
T: Escape character is '^]'.
C: GET /~uvalen/cas/index.html HTTP/1.0
C:
S: HTTP/1.1 200 OK
S: Date: Mon, 05 May 2003 17:47:49 GMT
S: Server: Apache/1.3.26 Ben-SSL/1.48 (Unix) Debian GNU/Linux mod_perl/1.26
S: Last-Modified: Mon, 05 May 2003 11:50:51 GMT
S: Etag: "6fc104-9797-3eb6501b"
S: Accept-Ranges: bytes
S: Content-Length: 38807
S: Connection: close
S: Content-Type: text/html
S:
S: <HTML>
…
S: </HTML>
T: Connection closed by foreign host.

La configuración de un servidor Web es algo mucho más complicado que la de un cliente Web, excediendo ampliamente los objetivos de este curso, por lo cual no se explica en estos apuntes.