5.- DESCRIPCIÓN DEL PROTOCOLO HTTP

5.1.- Introducción

El Protocolo de Transferencia de HiperTexto (Hypertext Transfer Protocol) es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes Web y los servidores HTTP. La especificación completa del protocolo HTTP 1/0 está recogida en el RFC 1945. Fue propuesto por Tim Berners-Lee, atendiendo a las necesidades de un sistema global de distribución de información como el World Wide Web.

5.2.- Características y funcionamiento

Desde el punto de vista de las comunicaciones, está soportado sobre los servicios de conexión TCP/IP, y funciona de la misma forma que el resto de los servicios comunes de los entornos UNIX: un proceso servidor escucha en un puerto de comunicaciones TCP (por defecto, el 80), y espera las solicitudes de conexión de los clientes Web. Una vez que se establece la conexión, el protocolo TCP se encarga de mantener la comunicación y garantizar un intercambio de datos libre de errores.

HTTP se basa en sencillas operaciones de solicitud / respuesta. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. El servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado. Todas las operaciones pueden adjuntar un objeto o recurso sobre el que actúan; cada objeto Web es conocido por su URL.

Los recursos u objetos que actúan como entrada o salida de un comando HTTP están clasificados por su descripción MIME. De esta forma, el protocolo puede intercambiar cualquier tipo de dato, sin preocuparse de su contenido. La transferencia se realiza en modo binario, byte a byte, y la identificación MIME permitirá que el receptor trate adecuadamente los datos.

Las principales características del protocolo HTTP son:

- Toda la comunicación entre los clientes y servidores se realiza a partir de caracteres de 8 bits. De esta forma, se puede transmitir cualquier tipo de documento: texto, binario, etc., respetando su formato original.

- Permite la transferencia de objetos multimedia. El contenido de cada objeto intercambiado está identificado por su clasificación MIME.

- Existen tres verbos básicos (hay más, pero por lo general no se utilizan) que un cliente puede utilizar para dialogar con el servidor: GET, para recoger un objeto, POST, para enviar información al servidor y HEAD, para solicitar las características de un objeto (por ej., la fecha de modificación de un documento HTML).

- Cada operación HTTP implica una conexión con el servidor, que es liberada al término de la misma. Es decir, en una operación se puede recoger un único objeto. En la actualidad se ha mejorado este procedimiento, permitiendo que una misma conexión se mantenga activa durante un cierto periodo de tiempo, de forma que sea utilizada en sucesivas transacciones. Este mecanismo, denominado HTTP Keep Alive, es empleado por la mayoría de los clientes y servidores modernos.

- No mantiene estado. Cada petición de un cliente a un servidor no es influida por las transacciones anteriores. El servidor trata cada petición como una operación totalmente independiente del resto.

- Cada objeto al que se aplican los verbos del protocolo está identificado a través de la información de situación del final de la URL.

Cada vez que un cliente realiza una petición a un servidor, se ejecutan los siguientes pasos:

1. Un usuario accede a una URL, seleccionando un enlace de un documento HTML o introduciéndola directamente en el campo Dirección.

2. El cliente Web descodifica la URL, separando sus diferentes partes. Así identifica el protocolo de acceso, la dirección DNS o IP del servidor, el posible puerto opcional (el valor por defecto es el puerto número 80) y el objeto requerido del servidor en la petición.

3. Se abre una conexión TCP/IP con el servidor, llamando al puerto TCP correspondiente en cada caso (el 80 por defecto, como comentamos).

4. Se realiza la petición. Para ello, se envía el comando necesario (GET, POST, HEAD,...), la dirección del objeto requerido (el contenido de la URL que sigue a la dirección del servidor), la

versión del protocolo HTTP empleada (casi siempre HTTP/1.0) y un conjunto variable de información, que incluye datos sobre las capacidades del navegador, datos opcionales para el servidor, etc.

5. El servidor devuelve la respuesta al cliente. Consiste en un código de estado y el tipo de dato MIME de la información de retorno, seguido de la propia información solicitada por el cliente.

6. Se cierra la conexión TCP.

Si no se utiliza el modo HTTP Keep Alive, este proceso se repite para cada acceso al servidor HTTP.

El diálogo con los servidores HTTP se establece a través de mensajes formados por líneas de texto, cada una de las cuales contiene los diferentes comandos y opciones del protocolo. Sólo existen dos tipos de mensajes, uno para realizar peticiones y otro para devolver la correspondiente respuesta.

La estructura general de los dos tipos de mensajes se puede ver en el siguiente esquema:

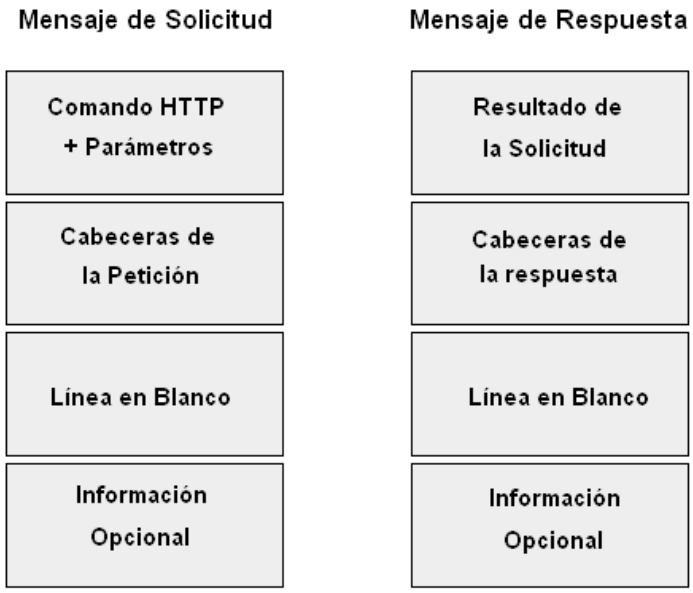

Figura 5-1: Estructura de mensajes de HTTP

La primera línea del mensaje de solicitud contiene el comando que se solicita al servidor HTTP, mientras que en la respuesta contiene el resultado de la operación que es un código numérico que permite conocer el éxito o fracaso de la operación. Después aparece, para ambos tipos de mensajes, un conjunto de cabeceras (unas obligatorias y otras opcionales), que condicionan y matizan el funcionamiento del protocolo.

La separación entre cada línea del mensaje se realiza con un par CR-LF (retorno de carro más nueva línea). El final de las cabeceras se indica con una línea en blanco, tras la cual se pueden incluir los datos transportados por el protocolo, por ejemplo, el documento HTML que devuelve un servidor.

5.3.- Comandos del protocolo HTTP

El protocolo HTTP consta de los siguientes comandos:

- **GET**, que se utiliza para recoger cualquier tipo de información del servidor. Se utiliza siempre que se pulsa sobre un enlace o se teclea directamente a una URL. Como resultado, el servidor HTTP envía el documento correspondiente a la URL. Este comando puede ir acompañado de una cabecera con los siguientes parámetros:
  - Accept: [*]/[*], que indica los formatos de archivo que puede soportar el cliente.
  - Accept-Charset: [formato]: indica los conjuntos de caracteres aceptados.
  - Accep-Encoding: [codificación]: Que tipo de codificación acepta.
  - Accept-Language: [lenguaje]: En que lenguaje se hará la respuesta que se proporcionará en la cabecera.
  - Authorization: [contraseña] [formato]: clave para tener acceso a lugares restringidos donde se requiere nombre de contraseña y el formato de autorización empleado para dicho proceso.
  - Conection: [opciones]: especifica opciones requeridas para una conexión.
  - Date: [fecha]: representa la fecha y la hora a la que se envió el comando.
  - From: [e-mail]: Campo opcional que incluye la dirección de correo electrónico particular del usuario concreto del cliente.
  - Host: [dirección del host]:([puerto]): Especifica la dirección del host del cliente y el puerto de conexión que ha empleado.
  - If-Modified-Since: [fecha]: Condición de que sólo se responda a la solicitud si la fecha de última modificación del objeto es

superior a la indicada en su parámetro.

- If-UnModified-Since: [fecha]: Condición de que sólo se responda a la solicitud si la fecha de última modificación del objeto no ha cambiado a partir de la fecha indicada en su parámetro.

- Pragma: [directivas]: Para incluir directivas de implementación.

- Proxy-Authorization: [autorización proxy]:[credenciales]: Identificarse a un proxy determinado.

- Referer: contiene la URL de la página que le ha dado acceso a la que está solicitando. Esto puede interesarle a los autores de páginas web para saber en que lugares existen enlaces de su página.

- Range: [rango]: establecer rango de bytes del contenido de la respuesta.

- Transfer-Enconding: [codificación]: indica el tipo de codificación aplicada del contenido que responderá el servidor.

- Upgrade: [protocolo]/[versión] : Especifica que protocolos suplementarios soporta el cliente, si también soportara FTP u otros.

- User-Agent: [cliente/versión] [(Sistema Operativo)]: Especifica cual es el nombre del cliente y su versión. El segundo parámetro corresponde al sistema operativo donde corre el cliente.

- Via: sirve para obtener la ruta de routers que ha recorrido el mensaje.

- **HEAD**, que es igual que el comando GET, y puede usar las mismas cabeceras, pero este comando le pide al servidor que le envíe solo como la respuesta la cabecera. Es utilizado por los gestores de cachés de páginas o los servidores proxy, para conocer cuándo es necesario actualizar la copia que se mantiene de un fichero.

- **POST,** que tiene un funcionamiento idéntico al comando HEAD, pero además, este comando envía datos de información al servidor, normalmente originaria de un formulario web para que el servidor los administre o los añada a una base de datos.

Posteriormente se han definido algunos comandos adicionales, que sólo están disponibles en determinadas versiones de servidores HTTP, con motivos eminentemente experimentales.

La última versión de HTTP, denominada 1.1, recoge estas y otras novedades, que se pueden utilizar, por ejemplo, para editar las páginas de un servidor Web trabajando en remoto.

Cuando el servidor envía la respuesta al cliente, le envía la información solicitada y una cabecera con una serie campos, estos campos son:

- Age: Tiempo transcurrido desde que se creó la respuesta.
- Allow: especifica que comandos puede utilizar el cliente respecto al objeto pedido, como pueden ser GET o HEAD.
- Expires: Fecha en la que caducará el objeto adjunto.
- Last-Modified: Fecha de la última modificación del objeto.
- Location: Se usa para redirigir la petición a otro objeto. Informa sobre la dirección exacta del objeto que se ha accedido.
- Proxy-Authenticate: Indica el esquema de autenticación en caso de que un proxy la requiera.
- Public: da una lista de los comandos que reconoce el servidor.
- Retry-After: si no puede ofrecer un objeto provisionalmente, indica la fecha para que se vuelva a hacer la petición.
- Server: Especifica el tipo y versión del servidor.
- Vary: Indica que hay mas de una posible respuesta a la petición pero solo escoge una.
- Warning: Específica información adicional sobre el estado de la respuesta.
- WWW-Autenticate: Usado cuando se accede a un lugar restringido, sirve para informar de las maneras de autentificarse que soporta el objeto del servidor.

Hay otros parámetros de información de cabeceras, que son válidos tanto para mensajes del cliente como para respuestas del servidor. Estas cabeceras son:

- Content-Type: descripción MIME de la información que contiene el mensaje para dar el tratamiento conveniente a los datos que se envían.

- Content-Length: Longitud en bytes de los datos enviados.

- Content-Encoding: Especifica en que formato están codificados los datos enviados.

- Date: Fecha local de la operación. Las fechas deben incluir la zona horaria en que reside el sistema que genera la operación. Por ejemplo: Sunday, 12-Dec-96 12:21:22 GMT+01. No existe un formato único en las fechas; incluso es posible encontrar casos en los que no se dispone de la zona horaria correspondiente, con los problemas de sincronización que esto produce. Los formatos de fecha a emplear están recogidos en los RFC 1036 y 1123.

- Pragma: Permite incluir información variada relacionada con el protocolo HTTP en el requerimiento o respuesta que se está realizando. Por ejemplo, un cliente envía un Pragma: no-cache para informar de que desea una copia nueva del recurso especificado.

Ante cada transacción con un servidor HTTP, éste devuelve un código numérico que informa sobre el resultado de la operación, como primera línea del mensaje de respuesta. Estos códigos aparecen en algunos casos en la pantalla del cliente, cuando se produce un error.

Los códigos de estados están clasificados en cinco categorías. La lista de códigos es la que podemos ver en la tabla siguiente:

Código	Descripción
2*??*	Operaciones realizadas satisfactoriamente
200	Todo correcto
201	Creado.
202	Aceptado
203	Información no autorizada
204	No hay contenido
205	Reiniciar contenido
206	Reiniciar contenido
3*??*	Mensajes de redirección.

301	Cambiado permanente
302	Cambiado temporalmente
303	Ir a otra
304	No modificado
305	Utilizar Proxy
307	Redirección Temporal
4*??*	Errores del cliente
400	Solicitud errónea
401	No autorizado
402	Se requiere pago.
403	Prohibido.
404	No encontrado.
405	Método no permitido.
406	No aceptable.
407	Se requiere autentificación de proxy.
408	El tiempo de la petición ha expedido.
409	Conflicto.
410	Ha desaparecido
411	Se requiere longitud.
412	Requiere longitud.
413	Entidad solicitada es demasiado grande.
414	El parámetro solicitado es demasiado grande.
415	Tipo de medio no soportado.
416	Rango de petición no complacible
417	Expectación fallida
5*??*	Errores del servidor
500	Error interno del servidor
501	No implementado

502	Gateway erróneo
503	Servicio no disponible
504	Algún router no ha recibido el mensaje a tiempo
505	Versión de http no soportado.

Figura 5-2: Códigos de estado de HTTP

El protocolo HTTP está muy extendido en el mundo de Internet, y cualquier usuario de Internet posee un navegador web, con el que se podrá conectar con el servidor web implementado sin tener que realizar ninguna otra operación que solicitar una página web como se hace normalmente. Así pues se ha optado por adoptar el protocolo HTTP para realizar y aceptar las peticiones de conexión a la aplicación que se va a desarrollar.