**UNIVERSIDAD TECNOLÓGICA CENTROAMERICANA**

**‘Investigación sobre el protocolo HTTP’**

**EXPERIENCIA DE USUARIO**

**Presentado a:**

**Ing. Elmer Danuary Padilla Sorto**

**Presentado Por:**

**Jasser Abdel Ramos Betancourt 11741013**

**Tegucigalpa, M. D. C., 28 de julio del 2020**

**El Protocolo HTTP**

El protocolo HTTP (HyperText Transfer Protocol) es quizá el protocolo más importante que debe conocer un desarrollador Web. El protocolo HTTP es un documento que define las reglas de los mensajes que se intercambian entre los navegadores (clientes) y los servidores.

**HTTP** es un protocolo cliente-servidor, lo que significa que el cliente envía una petición al servidor y espera un mensaje de respuesta.

Cuando abres el navegador e ingresas a cualquier URL los pasos que ocurren son los siguientes:

1. El navegador abre una conexión con el servidor.
2. El navegador envía un mensaje HTTP (a esto se le conoce como una petición).
3. El servidor recibe el mensaje, lo interpreta siguiendo las reglas del protocolo y responde con otro mensaje (la respuesta) que puede ser el documento que se solicitó el usuario, o un error (p.e. que el recurso no existe).
4. El navegador cierra la conexión.

**Mensajes HTTP**

Un mensaje HTTP (no importa si es de petición o respuesta) se compone de 3 partes:

* La primera línea (que es diferente para la petición y la respuesta).
* Encabezados.
* Una línea en blanco.
* El cuerpo (opcional)

**Ejemplo**:

1. GET /index.html HTTP/1.1
2. Host: wikipedia.org
3. Accept: text/html

Las principales **características del protocolo** **HTTP** son:

1. 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.
2. Permite la transferencia de objetos multimedia. El contenido de cada objeto intercambiado está identificado por su clasificación MIME.
3. 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 ejemplo, la fecha de modificación de un documento HTML).
4. 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.
5. 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.
6. 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 del cliente Web.
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 80) y el objeto requerido del servidor.
3. Se abre una conexión **TCP/IP** con el servidor, llamando al puerto TCP correspondiente.
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 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.
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**.