Ingeniería de Sistemas

Desarrollo y Servicios Web

Sesión 5

Fernando Barraza A.
fbarraza@puj.edu.co

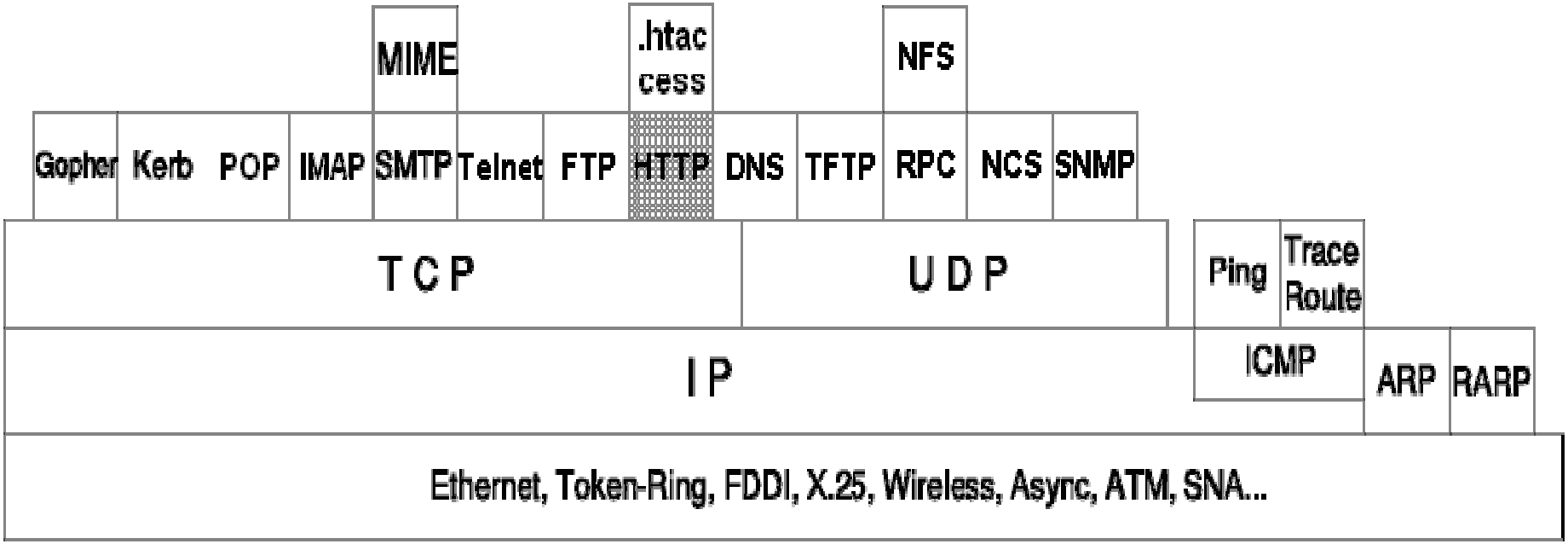
Sesión 5

- Objetivo: Entender la estructura y operación del protocolo HTTP
- Temas:
  - Estructura del protocolo
  - Métodos
  - Encabezados
  - Códigos

Que es HTTP?

- Es un protocolo de nivel de aplicación para sistemas de información hipermedia, distribuído, colaborativo
- Es un protocolo genérico, sin estados (stateless)
- A través de la extensión de sus métodos de petición, códigos de error y cabeceras puede usarse para multiples tareas diferentes al tratamiento de hipertexto como servidor de nombres o sistemas de gestión de objetos distribuídos.

Estructura del protocolo



Que permite HTTP?

- El tipado y negociación de la representación de los datos permite a los sistemas ser construídos independientemente de los datos que vayan a ser transferidos.
- Permite usar una serie de métodos para indicar la finalidad de la petición.

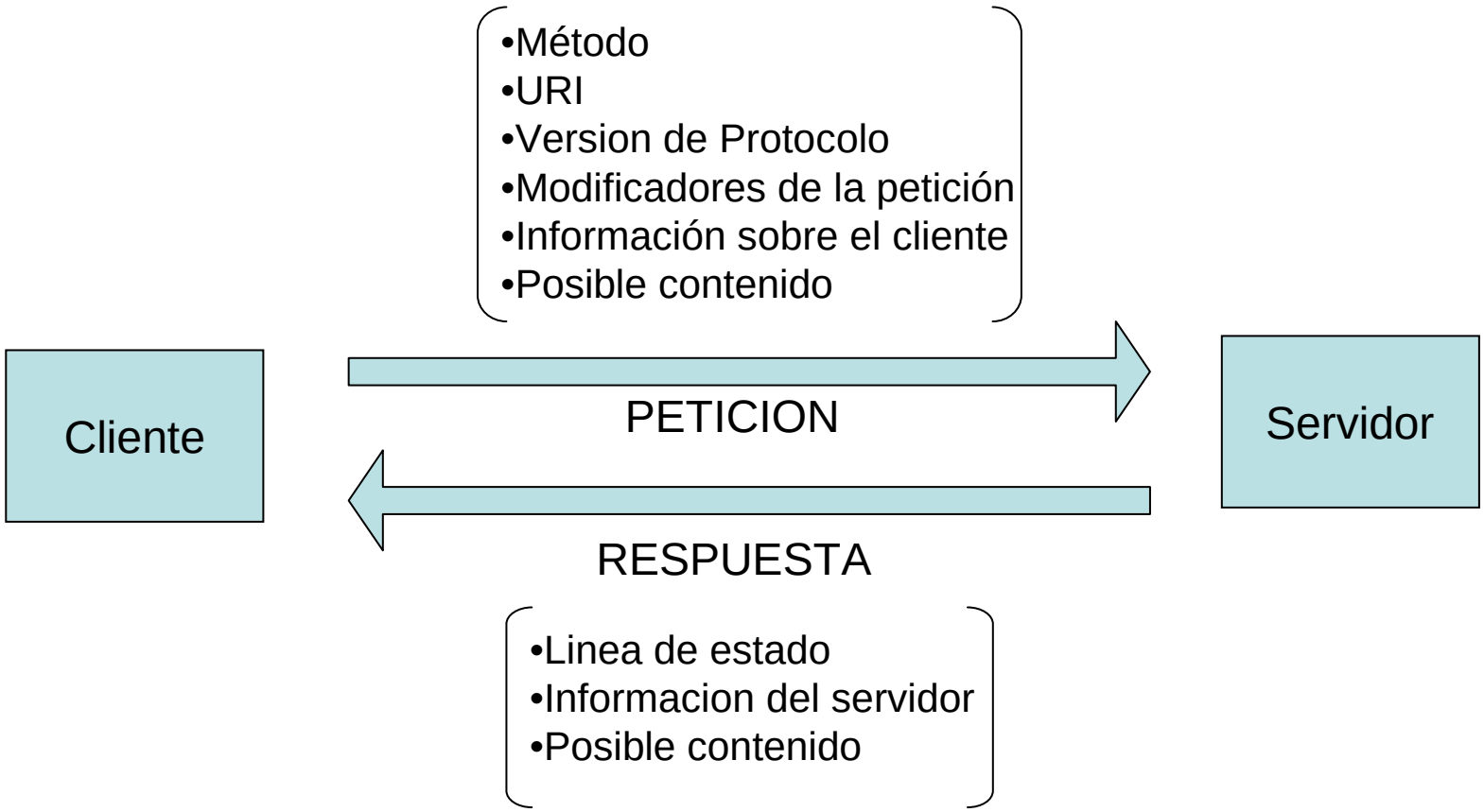
En que otros estándares se basa?

- Se basa en otros conceptos y estándares como *Uniform Resource Identifier* (URI), *Uniform Resource Location* (URL) y *Uniform Resource Name* (URN), para indicar el recurso al que hace referencia la petición.
- Los mensajes se pasan con un formato similar al usado por el *Internet Mail* y el *Multipurpose Internet Mail Extensions* (MIME).

Como funciona?

- El protocolo HTTP se basa en un paradigma de peticiones y respuestas.
- Un cliente inicia la comunicación envíando una petición y el servidor contesta con una respuesta, ambas en un formato predefinido.

Petición y respuesta



Caracteristicas

- 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 multimedia: texto, binario, etc., respetando su formato original.
- El contenido de cada objeto intercambiado está identificado por su clasificación MIME.

Caracteristicas (2)

- 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.
- 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.

Sintaxis HTTP

- La sintaxis de una petición es la siguiente:

http://dirección[:puerto]/[ruta]

- Donde *dirección* es el nombre de un dominio de Internet o una dirección IP, el *puerto* es un número que indica el puerto al que se envía la petición y la *ruta* indica el recurso al que se accede dentro del servidor.

Mensaje HTTP

- Consiste en una petición de un cliente al servidor y en la respuesta del servidor al cliente.
- Las peticiones y respuestas pueden ser simples o completas.
  - Completas: Se envían cabeceras y un contenido.
  - Simples: Sólo se puede usar el método GET y no hay contenido. Una respuesta simple sólo consta de contenido.

Estructura de una Petición

- Línea de petición

  Método (espacio) URI de la petición (espacio) Versión del protocolo CRLF

- *(Cabeceras)
- CRLF
- [Contenido]

Métodos

- GET *
- HEAD *
- POST
- PUT
- DELETE
- TRACE

*: Soportados para HTTP 0.9

Método GET

- Solicita la devolución de información al cliente identificada por la URI.
- Si la URI se refiere a un proceso que produce información, se devuelve la información y no la fuente del proceso.
- El método GET pasa a ser un GET condicional si la petición incluye las cabeceras If-Modified-Since, If-Unmodified-Since, If-Match, If-None-Match o If-Range. Estas cabeceras hacen que el contenido de la respuesta se transmita sólo si se cumplen unas condiciones determinadas por esas cabeceras. Esto se hico para reducir el tráfico en las redes.

Método HEAD

- Igual que el método GET, solo que el servidor no tiene que devolver el contenido, sólo las cabeceras.
- Este método se puede usar para obtener información sobre el contenido que se va a devolver en respuesta a la petición.
- Se suele usar también para chequear la validez de *links*, accesibilidad y modificaciones recientes.

Método POST

- Se usa para hacer peticiones en las que el servidor destino acepta el contenido de la petición como un nuevo subordinado del recurso pedido.
- Se creó para cubrir funciones como la de enviar un mensaje a grupos de usuarios, dar un bloque de datos como resultado de un formulario a un proceso de datos, añadir nuevos datos a una base de datos, ...
- La función llevada a cabo por el método POST está determinada por el servidor y suele depender de la URI de la petición.
- El resultado de la acción realizada por el método POST puede ser un recurso que no sea identificable mediante una URI.

Método PUT

- Permite guardar el contenido de la petición en el servidor bajo la URI de la petición.
  - Si la URI ya existe, entonces el servidor considera que esta petición proporciona una versión actualizada del recurso.
  - Si la URI indicada no existe y es válida para definir un nuevo recurso, el servidor puede crear el recurso con esa URI.
- La principal diferencia entre POST y PUT se encuentra en el significado de la URI. En el caso del método POST, la URI identifica el recurso que va a manejar en contenido, mientras que en el PUT identifica el contenido.

Cabeceras

- Generales: Aplican a peticiones y respuestas pero no al contenido
- De petición: Permiten al cliente pasar información al servidor sobre la petición y sobre el cliente.
- De respuesta: Información adicional del servidor al cliente sobre la respuesta dada.
- De entidad: Permiten definir información adicional sobre el contenido que se transmite (tipo) y en caso de que no haya contenido, sobre el recurso al que se quiere acceder con la petición.

Cabezeras generales

- Cache-Control
  - Son directivas que se han de tener en cuenta a la hora de mantener el contenido en una caché.
- Connection
  - Permite especificar opciones requeridas para una conexión.
- Date
  - Representa la fecha y la hora a la que se creó el mensaje.
- Pragma
  - Usado para incluir directivas de implementación.
- Transfer-Encoding
  - Indica la codificación aplicada al contenido.
- Upgrade
  - Permite al cliente especificar protocolos que soporta.
- Via
  - Usado por pasarelas y *proxies* para indicar los pasos seguidos.

Cabezeras de petición

- Accept
  - Indican el tipo de respuesta que acepta.
- Accept-Charset
  - Indica los conjuntos de caracteres que acepta.
- Accept-Encoding
  - Que tipo de codificación acepta.
- Accept-Language
  - Tipo de lenguaje de la respuesta que se prefiere.
- Authorization
  - El agente de usuario quiere autentificarse con el servidor.
- From
  - Contiene la dirección de correo que controla en agente de usuario.
- Host
  - Especifica la máquina y el puerto del recurso pedido.
- Proxy-Authorization
  - Permite que el cliente se identifique a un proxy.

Cabezeras de petición (2)

- If-Modified-Since
  - Para el GET condicional.
- If-Match
  - Para el GET condicional.
- If-None-Match
  - Para el GET condicional.
- If-Range
  - Para el GET condicional.
- If-Unmodified-Since
  - Para el GET condicional.
- Max-Forwards
  - Indica el máximo número de elementos por los que pasa.
- Range
  - Establece un rango de *bytes* del contenido.
- Referer
  - Indica la dirección donde obtuvo la URI de la petición.
- User-Agent
  - Información sobre el agente que genera la petición.

Cabeceras de respuesta

- Age
  - Estimación del tiempo transcurrido desde que se creó la respuesta.
- Location
  - Ee usa par a redirigir la petición a otra URI.
- Proxy-Authenticate
  - Ante una respuesta con el código 407 (autentificación *proxy* requerida), indica el esquema de autentificación.
- Public
  - Da la lista de métodos soportados por el servidor.
- Retry-After
  - Ante un servicio no disponible da una fecha para volver a intentarlo.
- Server
  - Información sobre el servidor que maneja las peticiones.
- Vary
  - Indica que hay varias respuestas y el servidor ha escogido una.
- Warning
  - Usada para aportar información adicional sobre el estado de la respuesta.
- WWW-Authenticate
  - Indica el esquema de autentificación y los parámetros aplicables a la URI.

Cabezeras de entidad

- Allow
  - Da los métodos soportados por el recurso designadopor la URI.
- Content-Base
  - Indica la URI base para resolver las URI relativas.
- Content-Encoding
  - Indica una codificación adicional aplicada al contenido (a parte de la aplicada por el tipo).
- Content-Language
  - Describe el idioma del contenido.
- Content-Length
  - Indica el tamaño del contenido del mensaje.
- Content-Location
  - Da información sobre la localización del recurso que da el contenido del mensaje.
- Content-MD5
  - Es un resumen en formato MD5 (RFC 1864) para chequear la integridad del contenido.
- Content-Range
  - En un GET parcial, indica la posición del contenido.
- Content-Type
  - Indica el tipo de contenido que es.
- Etag
  - Define una marca para el contenido asociado.
- Expires
  - Indica la fecha a partir de la cual la respuesta deja de ser válida.
- Last-Modified
  - Indica la fecha de la última modificación.

Estructura de una respuesta

- Línea de estado

  Versión del protocolo (espacio) Código de estado (espacio) Frase explicativa CRLF
- *(Cabeceras)
- CRLF
- [Contenido]

Codigos de estado

- El código de estado es un número de 3 dígitos que indica si la petición ha sido atendida satisfactoriamente o no, y en caso de no haber sido atendida, indica la causa.
- Los códigos se dividen en cinco clases definidas por el primer dígito del código de estado.

Clasificacion de códigos de estado

- 1xx:
  - Informativo. La petición se recibe y sigue el proceso. Esta familia de respuestas indican una respuesta provisional. Este tipo de respuesta está formada por la línea de estado y las cabeceras.
- 2xx:
  - Éxito. La acción requerida por la petición ha sido recibida, entendida y aceptada.
- 3xx:
  - Redirección. Para completar la petición se han de tomar más acciones.
- 4xx:
  - Error del cliente. La petición no es sintácticamente correcta y no se puede llevar a cabo.
- 5xx:
  - Error del servidor. El servidor falla al atender la petición que aparentemente es correcta.

Algunos códigos comunes

- 100: continuar.
- 200: éxito.
- 202: aceptado.
- 204: sin contenido.
- 206: contenido parcial.
- 305: usar *proxy*.
- 400: petición errónea.
- 401: no autorizado.
- 404: no encontrado.
- 405: método no permitido.
- 407: se requiere autentificación proxy.
- 408: límite de tiempo de la petición.
- 413: contenido de la petición muy largo.
- 500: error interno del servidor.
- 501: no implementado.
- 502: puerta de enlace errónea.
- 504: tiempo límite de la puerta de enlace.
- 505: versión de protocolo HTTP no soportada.

Créditos

- **Antonio Ocón Carreras**. Descripción técnica de TCP/IP. Universidad de las Palmas de Gran Canaria.