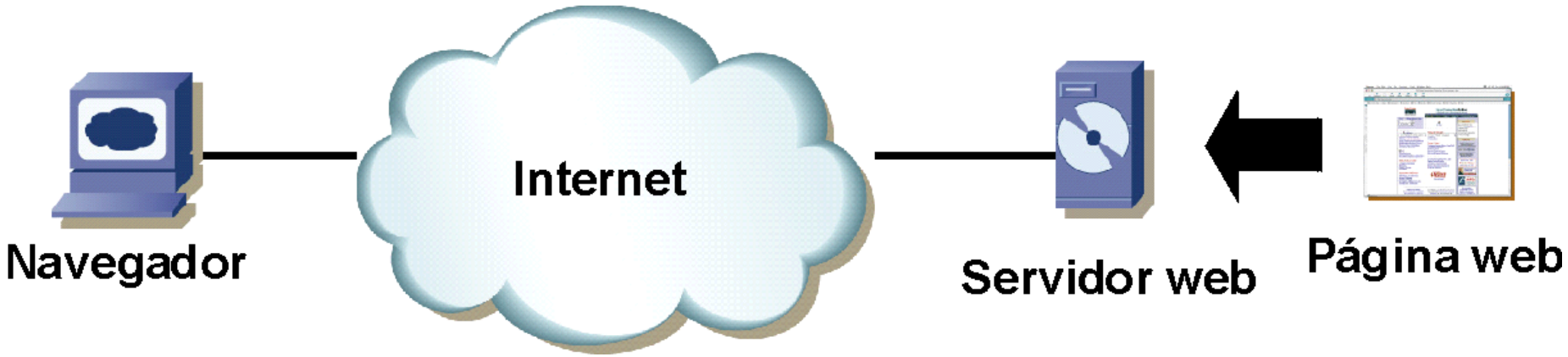
“Arquitectura de las Aplicaciones Web”

Rogelio Ferreira Escutia

Arquitectura General

Tecnología Web



- **Es toda la tecnología que tiene que ver con el manejo de información via internet (o a través de una intranet).**

Protocolo HTTP

HTTP

- **Hypertext Transfer Protocol o HTTP (en español protocolo de transferencia de hipertexto) es el protocolo usado en cada transacción de la World Wide Web.**

- **HTTP fué desarrollado por el World Wide Web Consortium y la Internet Engineering Task Force, colaboración que culminó en 1999 con la publicación de una serie de RFC, el más importante de ellos es el RFC 2616 que especifica la versión 1.1**

- **HTTP define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, proxies) para comunicarse.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

HTTP

- **Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.**
- **Al cliente que efectúa la petición (un navegador web o un spider) se lo conoce como "user agent" (agente del usuario).**
- **A la información transmitida se la llama recurso y se la identifica mediante un localizador uniforme de recursos (URL).**
- **Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento, etc.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

HTTP

- **HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores.**
- **El desarrollo de aplicaciones web necesita frecuentemente mantener estado.**
- **Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente.**
- **Esto le permite a las aplicaciones web instituir la noción de "sesión", y también permite rastrear usuarios ya que las cookies pueden guardarse en el cliente por tiempo indeterminado.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

HTTP

Hyper Text Transfer Protocol (HTTP)

Familia	Familia de protocolos de Internet
Función	Transferencia de hipertexto
Última versión	1.2
Puertos	80/TCP

Ubicación en la pila de protocolos

Aplicación	**HTTP**
Transporte	TCP
Red	IP

Estándares

RFC 1945 (HTTP/1.0, 1996)
RFC 2616 (HTTP/1.1, 1999)
RFC 2774 (HTTP/1.2, 2000)

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Sesión HTTP

Sesión HTTP

- **Una transacción HTTP está formada por un encabezado seguido, opcionalmente, por una línea en blanco y algún dato.**
- **El encabezado especificará cosas como la acción requerida del servidor, o el tipo de dato retornado, o el código de estado.**
- **El uso de campos de encabezados enviados en las transacciones HTTP le dan gran flexibilidad al protocolo.**
- **Estos campos permiten que se envíe información descriptiva en la transacción, permitiendo así la autenticación, cifrado e identificación de usuario.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Sesión HTTP (ejemplo)

Petición Cliente

- **Para obtener un recurso con el URL http://www.example.com/index.html, se abre una conexión al host www.example.com, puerto 80 que es el puerto por defecto para HTTP.**

- **Se envía un mensaje en el estilo siguiente:**

```
GET /index.html HTTP/1.1
Host: www.example.com
User-Agent: nombre-cliente
[Línea en blanco]
```

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Sesión HTTP (ejemplo)

Respuesta Servidor

- **La respuesta del servidor está formada por encabezados seguidos del recurso solicitado, en el caso de una página web sería lo siguiente:**

```
HTTP/1.1 200 OK
Date: Fri, 31 Dec 2003 23:59:59 GMT
Content-Type: text/html
Content-Length: 1221

<html>
<body>
<h1>Página principal de tuHost</h1>
(Contenido)
  .
  .
  .
</body>
</html>
```

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

Métodos HTTP

- **HTTP define 8 métodos (algunas veces referido como "verbos") que indica la acción que desea que se efectúe sobre el recurso identificado.**

- **Lo que este recurso representa, si los datos pre-existentes o datos que se generan de forma dinámica, depende de la aplicación del servidor.**

- **A menudo, el recurso corresponde a un archivo o la salida de un ejecutable que residen en el servidor.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

HEAD

- **Pide una respuesta idéntica a la que correspondería a una petición GET, pero sin el cuerpo de la respuesta. Esto es útil para la recuperación de meta-información escrita en los encabezados de respuesta, sin tener que transportar todo el contenido.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

GET

- **Pide una representación del recurso especificado. Por seguridad no debería ser usado por aplicaciones que causen efectos ya que transmite información a través de la URI agregando parámetros a la URL.**

- **Ejemplo:**

GET /images/logo.png HTTP/1.1
obtiene un recurso llamado logo.png

Ejemplo con parámetros:
/index.php?page=main&lang=es

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

POST

- **Somete los datos a que sean procesados para el recurso identificado.**
- **Los datos se incluirán en el cuerpo de la petición.**
- **Esto puede resultar en la creación de un nuevo recurso o de las actualizaciones de los recursos existentes o ambas cosas.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

PUT

- **Sube, carga o realiza un upload de un recurso especificado (archivo), es el camino más eficiente para subir archivos a un servidor, esto es porque en POST utiliza un mensaje multiparte y el mensaje es decodificado por el servidor.**
- **En contraste, el método PUT te permite escribir un archivo en una conexión socket establecida con el servidor.**
- **La desventaja del método PUT es que los servidores de hosting compartido no lo tienen habilitado.**
- **Ejemplo:**
  PUT /path/filename.html HTTP/1.1

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

DELETE

- **Borra el recurso especificado.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

TRACE

- **Este método solicita al servidor que envíe de vuelta en un mensaje de respuesta, en la sección del cuerpo de entidad, toda la data que reciba del mensaje de solicitud.**

- **Se utiliza con fines de comprobación y diagnóstico.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

OPTIONS

- **Devuelve los métodos HTTP que el servidor soporta para un URL específico.**
- **Esto puede ser utilizado para comprobar la funcionalidad de un servidor web mediante petición en lugar de un recurso específico.**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Métodos HTTP

CONNECT

- **Se utiliza para saber si se tiene acceso a un host, no necesariamente la petición llega al servidor, este método se utiliza pricipalmente para saber si un proxy nos da acceso a un host bajo condiciones especiales, como por ejemplo "corrientes" de datos bidireccionales encriptadas (como lo requiere SSL).**

"Hipertext Transfer Protocol", http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol, marzo 2013

Códigos de Estado HTTP

Mensajes del HTTP

- **Cuando se realiza una petición utilizando un método HTTP del Cliente al Servidor, se generan códigos de estado (mensajes) de acuerdo al resultado de la petición (se hizo, no se hizo, ocurrió algún error o excepción).**

- **La lista completa se los Códigos de Estado se encuentran en el RFC 2616 y algunos otros en los estándares RFC 2518, RFC 2817, RFC 2295, RFC 2774 y RFC 4918 .**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Clasificación

- Los códigos de estado son números de 3 cifras.
- Se clasifican de acuerdo al tipo al tipo de evento que se generó.
- El primer número nos indica la sección a la cual pertenece el código estado y se dividen en 5 seciones principales (del 100 al 500). No todos existen y no todos están implementados.
- Las secciones son las siguientes:
  - 1xx: Respuestas informativas
  - 2xx: Peticiones correctas
  - 3xx: Redirecciones
  - 4xx Errores del cliente
  - 5xx Errores de servidor

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (1)

2XX Peticiones Correctas

- **Esta clase de código de estado indica que la petición fue recibida correctamente, entendida y aceptada.**
- **200 OK - Respuesta estándar para peticiones correctas.**
- **201 Creado - La petición ha sido completada y ha resultado en la creación de un nuevo recurso.**
- **202 Aceptada - La petición ha sido aceptada para procesamiento, pero este no ha sido completado. La petición eventualmente pudiere no ser satisfecha, ya que podría ser no permitida o prohibida cuando el procesamiento tenga lugar.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (2)

2XX Peticiones Correctas

- 203 Información no autoritativa (desde HTTP/1.1)
- 204 Sin contenido
- 205 Recargar contenido
- 206 Contenido parcial

  La petición servirá parcialmente el contenido solicitado. Esta característica es utilizada por herramientas de descarga como wget para continuar la transferencia de descargas anteriormente interrumpidas, o para dividir una descarga y procesar las partes simultáneamente.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (3)

2XX Peticiones Correctas

- **207 Estado múltiple (Multi-Status, WebDAV)**
  El cuerpo del mensaje que sigue es un mensaje XML y puede contener algún número de códigos de respuesta separados, dependiendo de cuántas sub-peticiones sean hechas.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (4)

3XX Redirecciones

- El cliente tiene que tomar una acción adicional para completar la petición.
- Esta clase de código de estado indica que una acción subsecuente necesita efectuarse por el agente de usuario para completar la petición.
- La acción requerida puede ser llevada a cabo por el agente de usuario sin interacción con el usuario si y solo si el método utilizado en la segunda petición es GET o HEAD. El agente de usuario no debe redirigir automáticamente una petición más de 5 veces, dado que tal funcionamiento indica usualmente un Bucle infinito.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (5)

3XX Redirecciones

- **300 Múltiples opciones**
  Indica opciones múltiples para el URI que el cliente podría seguir. Esto podría ser utilizado, por ejemplo, para presentar distintas opciones de formato para video, listar archivos con distintas extensiones o word sense disambiguation.

- **301 Movido permanentemente**
  Esta y todas las peticiones futuras deberían ser dirigidas a la URI dada.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (6)

3XX Redirecciones

- **302 Movido temporalmente**

Este es el código de redirección más popular, pero también un ejemplo de las prácticas de la industria contradiciendo el estándar. La especificación HTTP/1.0 (RFC 1945) requería que el cliente realizara una redirección temporal (la frase descriptiva original fue "Moved Temporarily"), pero los navegadores populares lo implementaron como 303.

Por tanto, HTTP/1.1 añadió códigos de estado 303 y 307 para eliminar la ambigüedad entre ambos comportamientos. Sin embargo, la mayoría de aplicaciones web y bibliotecas de desarrollo aún utilizan el código de respuesta 302 como si fuera el 303.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (7)

3XX Redirecciones

- **303 (desde HTTP/1.1)**
  La respuesta a la petición puede ser encontrada bajo otra URI utilizando el método GET.

- **304 No modificado**
  Indica que la petición a la URL no ha sido modificada desde que fue requerida por última vez. Típicamente, el cliente HTTP provee un encabezado como If-Modified-Since para indicar una fecha y hora contra la cual el servidor pueda comparar. El uso de este encabezado ahorra ancho de banda y reprocesamiento tanto del servidor como del cliente.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (8)

3XX Redirecciones

- **305 Utilice un proxy (desde HTTP/1.1)**
  Muchos clientes HTTP (como Mozilla2 e Internet Explorer) no se apegan al estándar al procesar respuestas con este código, principalmente por motivos de seguridad.

- **306 Cambie de proxy**
  Esta respuesta está descontinuada.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (9)

3XX Redirecciones

- **307 Redirección temporal (desde HTTP/1.1)**

  Se trata de una redirección que debería haber sido hecha con otra URI, sin embargo aún puede ser procesada con la URI proporcionada. En contraste con el código 303, el método de la petición no debería ser cambiado cuando el cliente repita la solicitud. Por ejemplo, una solicitud POST tiene que ser repetida utilizando otra petición POST.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (10)

4XX Errores del Cliente

- **La solicitud contiene sintaxis incorrecta o no puede procesarse.**
- **La intención de la clase de códigos de respuesta 4xx es para casos en los cuales el cliente parece haber errado la petición.**
- **Excepto cuando se responde a una petición HEAD, el servidor debe incluir una entidad que contenga una explicación a la situación de error, y si es una condición temporal o permanente. Estos códigos de estado son aplicables a cualquier método de solicitud (como GET o POST). Los agentes de usuario deben desplegar cualquier entidad al usuario. Estos son típicamente los códigos de respuesta de error más comúnmente encontrados.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (11)

4XX Errores del Cliente

- **400 Solicitud incorrecta**
  - **La solicitud contiene sintaxis errónea y no debería repetirse.**

- **401 No autorizado**
  - **Similar al 403 Forbidden, pero específicamente para su uso cuando la autentificación es posible pero ha fallado o aún no ha sido provista. Vea autentificación HTTP básica y Digest access authentication.**

- **402 Pago requerido**
  - **La intención original era que este código pudiese ser usado como parte de alguna forma o esquema de Dinero electrónico o micropagos, pero eso no sucedió, y este código nunca se utilizó.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (12)

4XX Errores del Cliente

- **403 Prohibido**
  - **La solicitud fue legal, pero el servidor se rehúsa a responderla. En contraste a una respuesta 401 No autorizado, la autentificación no haría la diferencia.**

- **404 No encontrado**
  - **Recurso no encontrado. Se utiliza cuando el servidor web no encuentra la página o recurso solicitado.**

- **405 Método no permitido**
  - **Una petición fue hecha a una URI utilizando un método de solicitud no soportado por dicha URI; por ejemplo, cuando se utiliza GET en una forma que requiere que los datos sean presentados vía POST, o utilizando PUT en un recurso de solo lectura.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (13)

4XX Errores del Cliente

- **406 No aceptable**
  - **El servidor no es capaz de devolver los datos en ninguno de los formatos aceptados por el cliente, indicados por éste en la cabecera "Accept" de la petición.**
- **407 Autenticación Proxy requerida**
- **408 Tiempo de espera agotado**
  - **El cliente falló al continuar la petición - excepto durante la ejecución de videos Adobe Flash cuando solo significa que el usuario cerró la ventana de video o se movió a otro. ref**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (14)

4XX Errores del Cliente

- **409 Conflicto**
  - **Indica que la solicitud no pudo ser procesada debido a un conflicto con el estado actual del recurso que esta identifica.**

- **410 Ya no disponible**
  - **Indica que el recurso solicitado ya no está disponible y no lo estará de nuevo. Debería ser utilizado cuando un recurso ha sido quitado de forma permanente.**

  - **Si un cliente recibe este código no debería volver a solicitar el recurso en el futuro. Por ejemplo un buscador lo eliminará de sus índices y lo hará más rápidamente que utilizando un código 404.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (15)

4XX Errores del Cliente

- 411 Requiere longitud
- 412 Falló precondición
- 413 Solicitud demasiado larga
- 414 URI demasiado larga
- 415 Tipo de medio no soportado
- 416 Rango solicitado no disponible
- El cliente ha preguntado por una parte de un archivo, pero el servidor no puede proporcionar esa parte, por ejemplo, si el cliente preguntó por una parte de un archivo que está más allá de los límites del fin del archivo.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (16)

4XX Errores del Cliente

- 417 Falló expectativa
- 418 "I´m a teapot" o "Soy una tetera".
- 422 Entidad no procesable (WebDAV - RFC 4918)
  - La solicitud está bien formada pero fue imposible seguirla debido a errores semánticos.
- 423 Bloqueado (WebDAV - RFC 4918)
  - El recurso al que se está teniendo acceso está bloqueado.
- 424 Falló dependencia (WebDAV) (RFC 4918)
  - La solicitud falló debido a una falla en la solicitud previa.

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (17)

4XX Errores del Cliente

- **425 Colección sin ordenar**
  - **Definido en los drafts de WebDav Advanced Collections, pero no está presente en "Web Distributed Authoring and Versioning (WebDAV) Ordered Collections Protocol" (RFC 3648).**

- **426 Actualización requerida (RFC 2817)**
  - **El cliente debería cambiarse a TLS/1.0.**

- **429 Demasiadas peticiones**
  - **Hay muchas conexiones desde esta dirección de internet**

- **449 Reintente conUna extensión de Microsoft**
  - **La petición debería ser reintentada después de hacer la acción apropiada.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (18)

5XX Errores de Servidor

- **El servidor falló al completar una solicitud aparentemente válida.**

- **Los códigos de respuesta que comienzan con el dígito "5" indican casos en los cuales el servidor tiene registrado aún antes de servir la solicitud, que está errado o es incapaz de ejecutar la petición. Excepto cuando está respondiendo a un método HEAD, el servidor debe incluir una entidad que contenga una explicación de la situación de error, y si es una condición temporal o permanente.**

- **Los agentes de usuario deben desplegar cualquier entidad incluida al usuario. Estos códigos de repuesta son aplicables a cualquier método de petición.**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (19)

5XX Errores de Servidor

- **500 Error interno**
  - **Es un código comúnmente emitido por aplicaciones empotradas en servidores web, mismas que generan contenido dinámicamente, por ejemplo aplicaciones montadas en IIS o Tomcat, cuando se encuentran con situaciones de error ajenas a la naturaleza del servidor web.**
- **501 No implementado**
- **502 Pasarela incorrecta**
- **503 Servicio no disponible**

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Códigos de Estado HTTP (20)

5XX Errores de Servidor

- 504 Tiempo de espera de la pasarela agotado
- 505 Versión de HTTP no soportada
- 506 Variante también negocia (RFC 2295)
- 507 Almacenamiento insuficiente (WebDAV - RFC 4918)
- 509 Límite de ancho de banda excedido
  - Este código de estatus, a pesar de ser utilizado por muchos servidores, no es oficial.
- 510 No extendido (RFC 2774)

"Códigos de Estado HTTP", http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP, febrero 2015

Rogelio Ferreira Escutia

Instituto Tecnológico de Morelia
Departamento de Sistemas y Computación

Correo:	***rogeplus@gmail.com***
	rferreir@itmorelia.edu.mx
Página Web:	***http://antares.itmorelia.edu.mx/~kaos/***
	http://www.xumarhu.net/
Twitter:	***http://twitter.com/rogeplus***
Facebook:	***http://www.facebook.com/groups/xumarhu.net/***