UNIVERSITAT OBERTA DE CATALUNYA

Ingeniería Técnica Informática Sistemas

ÁREA REDES DE COMPUTADORES

# CREACIÓN DEL NÚCLEO DE UN SERVIDOR WEB

**Alumno:** Ángel del Río Medina

**Dirigido por:** Jesús Arribi Vilela

**Curso**: 2004-2005

UOC

La universidad virtual

Jesús Arribi Vilela

Tutor UOC - Informática de sistemas
Consultor FC – Postgrado
Consultor UOC - Informática de sistemas
Colaborador Docente
Colaborador – UOC
Consultor UOC - Informática de gestión

ÁNGEL Del Río Medina

Ingeniero Técnico informática de gestión
Estudiante - Informática de sistemas
Estudiante -Ingeniería Informática
Simpatizante – UOC

**TFC –Área Redes de computadores**
**Alumno:** Ángel del Río Medina
**Dirigido por:** Jesús Arribi Vilela
**Curso**: 2004-2005

## Índice





## 1. Dedicatoria y agradecimientos

Este trabajo está dedicado a mi esposa María Victoria, por su paciencia y comprensión conmigo durante estos interminables años de carrera.

También quiero agradecer el cariño y el apoyo de mi hermana y de mis padres. A los cuales también dedico este trabajo.

# 2. Presentación

## 2.1. Descripción[1]

El trabajo fin de carrera (TFC) es una asignatura que está pensada para realizar un trabajo de síntesis de los conocimientos adquiridos en otras asignaturas de la carrera y que requiere ponerlos en práctica conjuntamente en un trabajo concreto. Normalmente el TFC es un trabajo eminentemente práctico y vinculado al ejercicio profesional de la informática aunque en algunos casos puede ser, o incluir, un trabajo de investigación.

[1] Extracto del Área de Redes de computadores de los TFC del año 2004-2005

# 3. Objetivos

## 3.1. Objetivos generales:

En esta área de Trabajos de Final de Carrera, se pretende que el estudiante desarrolle un proyecto de Informática relacionado estrechamente con el área de las Redes de Computadores. Esto implica que no sólo hará falta que se apliquen conceptos básicos de desarrollo de proyectos de informática adquiridos durante los estudios, si no que un esfuerzo importante deberá estar dedicado a la resolución de alguna problemática inherente a las redes de computadores. Esta aproximación al área de las redes se podrá hacer a diferentes niveles, según los intereses del estudiante, cubriendo desde los niveles más bajos de las pilas de protocolos hasta las aplicaciones basadas en redes más extendidas actualmente.

El desarrollo de un proyecto de estas características tendría que cubrir de manera más o menos detallada los siguientes puntos:

- Estudio de la situación actual
- Evaluación de la utilidad
- Evaluación de la usabilidad (educación, empresas públicas / privadas)
- Diseño de la aplicación
- Implementación
- Bibliografía y trabajos relacionados

Así pues, en esta área, se pretende diseñar e implementar una serie de proyectos que estén directamente conectados con algún aspecto de las redes de computadores y que a la vez sean justificables para solucionar alguna problemática existente o para ofrecer algún servicio requerido.
En este caso concreto se ha elegido el desarrollo de un servidor web.

## 3.2. Objetivos particulares:

Una vez realizados los objetivos generales, el objetivo particular es realizar el núcleo principal de un servidor web, cuyas características son:

- Multithread
- Multiproceso
- Desarrollado en Java
- Funciona en entornos Windows, GNU Linux y heterogéneos.

## 4. Tareas a realizar

Las tareas hasta obtener la memoria son las siguientes

### Tarea1: Plan de trabajo (PEC1)

**Temporización:** 2 semanas (16 de septiembre al 29 de septiembre).

Elaboración de un documento en el cual estén identificadas y descritas las tareas necesarias para llevar a cabo el proyecto y su distribución de tiempos. Este documento deberá ser capaz de responder cuestiones como: ¿Qué se quiere hacer? ¿Qué aspira tener cuando finalice el trabajo? ¿Qué se espera encontrar? ¿Cómo lograr el objetivo general? ¿Cómo se logran los objetivos específicos?

### Tarea 2: Recogida y clasificación de toda la información relevante (PEC2)

**Temporización:** 2 semanas (29 de septiembre al 12 de octubre).

Recogida y clasificación de toda la información que pueda ser relevante para llevar a cabo el proyecto:

- Temas relacionados con el protocolo http: RFCs, etc.
- Temas relacionados con los servidores web existentes.
- Temas relacionados con el análisis, el diseño y la implementación de software cliente / servidor.
-

**Objetivos:**

- Profundizar los conocimientos sobre el protocolo http.
- Tener una visión clara sobre la funcionalidad que tendrá que tener el servidor web.
- Tener una visión general sobre los servidores web más usados del momento.

**Hitos:**

- Documento descriptivo del protocolo http.
- Crónica sobre servidores web existentes.

### Tarea 3: Análisis y diseño del servidor a implementar. (PEC2)

**Temporización:** 4 semanas (13 de octubre al 9 de noviembre).

Análisis y diseño del servidor a implementar.

**Objetivos:**

- Establecer requisitos que tendrá que tener el servidor.
- Obtener el análisis del software en función de los requisitos y de la funcionalidad que se le quiere dar.



- Obtener el diseño de la aplicación a implementarla basándonos en el análisis del punto anterior.

**Hitos:**
- Obtener el análisis y el diseño de la aplicación documentando las decisiones tomadas.

### Tarea 4: Implementación del servidor y juego de pruebas. (PEC3)

**Temporización:** 6 semanas (10 de noviembre al 21 de diciembre).

**Objetivos:**
- Implementar el servidor en función del diseño obtenido.
- Generar los juegos de pruebas adecuadas para garantizar el correcto funcionamiento del servidor, y para comprobar que se cubren los requisitos establecidos.

**Hitos:**
- Creación del servidor web.
- Documento sobre la implementación (decisiones, técnicas utilizadas como *sockets, pipes, thread*s, etc.).
- Documentación sobre los juegos de pruebas.

### Tarea 5: Documentación del producto.

**Temporizació**n: 1 semana (del 22 al 28 de diciembre).

Documentación del producto

**Objetivos:**
- Documentar la instalación del servidor web.
- Documentar el uso de las diferentes opciones de configuración que pueda tener.

**Hitos:**
- Documentación del servidor web.

### Tarea 6: Arreglos finales de la memoria y preparar la presentación del servidor web que se ha creado.

**Temporización:** 2 semanas (del 29 de diciembre al 9 de enero).

Arreglos finales de la memoria y preparar la presentación del servidor web que se ha creado.

**Objetivos:**
- Síntesis de la memoria tomada durante el proyecto.
- Crear una presentación del servidor web creado.

**Hitos:**



- Memoria del proyecto.
- Presentación del servidor web.

# 5. Productos

## 5.1. Productos

Los productos a obtener son tres:

**Memoria del proyecto**, (este documento) que sintetiza el trabajo realizado en el TFC y que demuestra que se han alcanzado los objetivos propuestos.

**Software del servidor web y juego de pruebas**, software con el que se consolidan todos los pasos realizados en la memoria y se verifica su funcionamiento.

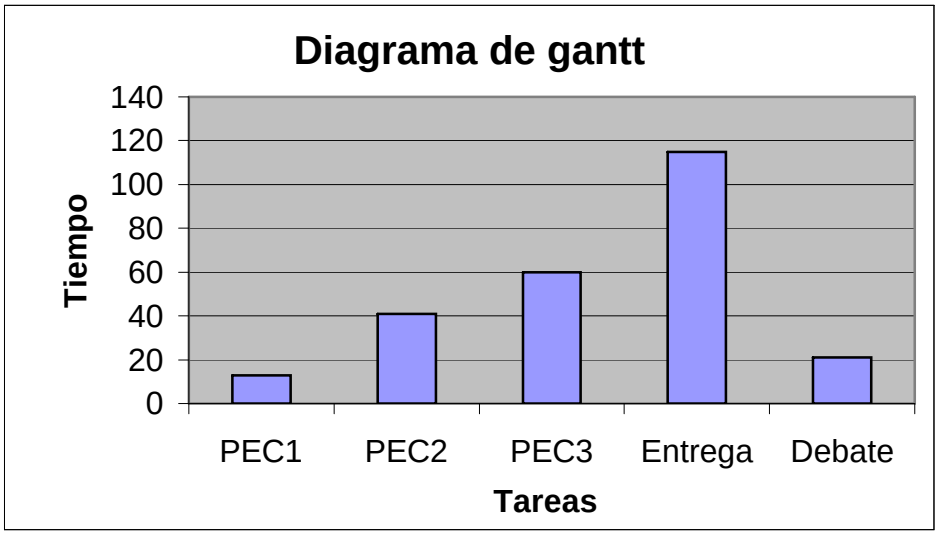
**Presentación virtual**, que sintetiza el trabajo realizado a lo largo del semestre y los resultados obtenidos.

# 6. Calendario

## 6.1. Calendario tabular con las tareas

| Nombre | Inicio | Entrega |
|---|---|---|
| PEC1 | 16/09/2004 | 29/09/2004 |
| PEC2 | 29/09/2004 | 09/11/2004 |
| PEC3 | 10/11/2004 | 09/01/2005 |
| Entrega Final: Memoria y Presentación Virtual | 16/09/2005 | 09/01/2005 |
| Cierre Debate Virtual | 09/01/2005 | 30/01/2005 |

## 6.2. Diagrama de Gantt



# 7. Introducción

## 7.1 ¿Qué es Internet?

Internet es una red global de equipos que se comunican mediante lenguajes y protocolos comunes a través de las líneas telefónicas.

Un servicio fundamental de la red en Internet es World Wide Web (WWW o Web). Los usuarios de Internet pueden crear páginas Web vinculadas entre sí por medio del Protocolo de transferencia de hipertexto (HTTP, *Hypertext Transfer Protocol*). Todas las páginas Web, incluidas las páginas principales de los sitios Web, tienen una dirección única llamada Ubicador uniforme de recursos (URL, *Uniform Resource Locator*). Éste es un ejemplo de URL: http://www.microsoft.com/spain/ntserver/

Las páginas Web son documentos de hipertexto (archivos a los que se ha dado formato mediante el Lenguaje de marcado de hipertexto (HTML, *Hypertext Markup Language*) que contienen hipervínculos. Los hipervínculos tienen direcciones Web incrustadas en ellos y aparecen como palabras subrayadas, y gráficos y palabras bordeados. Cuando se hace clic en un hipervínculo, el usuario "salta" a la ubicación de Internet especificada en el hipervínculo.

Los servidores Web proporcionan automáticamente texto con formato, gráficos, sonidos y animaciones a los usuarios de Internet. Para conectar con servidores Web y ver su información, se necesita un explorador Web, como Microsoft Internet Explorer.

## 7.2 ¿Qué es una intranet?

Las intranets existen a nivel local y consisten en equipos que están conectados por medio de redes LAN. Internet y las intranets privadas comparten muchas características, pero también tienen diferencias notables. Una intranet es una red interna dentro de una organización que utiliza la tecnología de Internet, como los servidores HTTP y los servicios de exploración de Web, para mejorar las comunicaciones internas, la publicación de información o el proceso de desarrollo de aplicaciones. Denominaremos intranet a cualquier red basada en TCP/IP que utilice tecnologías de Internet y no esté conectada a Internet.

## 7.3 Consideraciones sobre seguridad

Internet, al igual que otras redes, proporciona comunicaciones bidireccionales. Cuando un usuario establece una conexión con Internet, otros equipos pueden ver el equipo del usuario. Algunos sistemas operativos, por ejemplo Windows NT, protege a los equipos frente a posibles intromisiones en el sistema.



### 7.3.1 Integración de un sitio intranet en Internet

Es posible integrar una intranet de empresa en Internet. Ambas redes pueden ser compatibles con el mismo sistema de red. Se deben tener en cuenta las siguientes implicaciones de seguridad antes de intentar integrar una intranet con Internet:

- Mantenga los datos descargados en un sitio intranet separados de la información distribuida a través de Internet. Por ejemplo, en las intranets se suelen distribuir documentos que podrían desvelar secretos comerciales o perjudicar a la compañía si estuvieran disponibles en Internet.

- Generalmente, los sitios intranet son sencillos e informales, mientras que los sitios Internet suelen reflejar la imagen pública de la organización.

- Puede ser desaconsejable otorgar derechos completos de acceso a una intranet a los usuarios de Internet.

# 8. Servidores Web

## 8.1 ¿Qué es un servidor web?

Un servidor web es un programa de aplicación que satisface las solicitudes HTTP realizadas por los navegadores. Para ello, el ordenador que la soporta debe estar conectado a la Internet y, por lo tanto, ha de tener asignada una dirección IP.


Un servidor web debe soportar los protocolos estándar en la Internet. Por ejemplo HTTP (protocolo de transferencia de hipertexto) que facilita el intercambio de datos entre el servidor web y el navegador. Además, para publicar una página se suele utilizar un protocolo más antiguo, el FTP (Protocolo de transferencia de archivos).

Adicionalmente, deben ofrecer soporte a scripts y aplicaciones en los lenguajes más comunes utilizados en aplicaciones de Internet, como Java, PHP y otros. Finalmente, debe contener algunos elementos de seguridad.

Los navegadores, por su parte, pueden recibir archivos mediante HTTP y FTP y poseer capacidad para interpretar scripts en lenguajes con Java y Javascript.

## 8.2 Tipos de servidores Web

Los dos servidores de la red más difundidos son, sin duda alguna, IIS de Microsoft y el adversario Apache, nacido de las cenizas de un anterior proyecto y de un grupo de voluntarios.

### IIS

El servidor Internet Information Server, creado por la empresa Microsoft, es un servidor de páginas Web, FTP y Gopher que soporta los protocolos HTTP, FTP y GOPHER.

Este servidor tiene varias características interesantes:

- Facilidad de instalación
- Utiliza una herramienta de configuración gráfica, lo que ayuda en las labores de administración y seguridad del servidor.
- Permite la administración remota vía Web, utilizando para ello un navegador (por ejemplo, el Internet Explorer).
- Está completamente integrado con el sistema de seguridad de Windows NT. Los usuarios acceden a las páginas en función de sus permisos sobre ellas.
- Permite la publicación mediante herramientas Web, como el FrontPage y similares, o mediante la simple copia de ficheros sobre la red.
- Permite utilizar mecanismos de seguridad avanzados, como Autentificación cifrada de NT.
- Permite integrar de un modo sencillo el servidor SQL de Microsoft. Se pueden utilizar varios mecanismos para crear consultas a las bases de datos (idc/htx, dbweb y otros).
- Se pueden crear páginas activas y scripts, soportando Active Server páges (ASP), protocolos CGI y ISAPI, que añaden dinamismo a las páginas.

**NCSA**

Era el más expandido hasta 1996. Numerosas informaciones y muy bien representadas están disponibles sobre su instalación, su configuración, etc.

**Apache**

Desde 1996, es el servidor más expandido. Actualmente, más de 70% de los servidores WWW en el mundo utilizan el programa Apache (porcentaje en continuo aumento). Al nivel de la configuración, es un servidor enteramente compatible con el servidor NCSA, pero más rápido. Por lo tanto es muy fácil migrar del servidor NCSA hacia el servidor Apache. Para obtener mejores rendimientos, para responder a un requerimiento desde que éste llega, mientras que el servidor NCSA crea un proceso en el momento en el cual un requerimiento es recibido.
Permite utilizar los mismos scripts CGI que con el servidor NCSA.
Esta disponible en Linux, aunque desde 1998, existen versiones para plataformas Microsoft. Un interfaz para OpenSSL permite añadir un mecanismo de encriptación (SSL y TLS) garantizando la confidencialidad y la seguridad de la comunicación entre servidor y navegador.

**CERN**

Es el primer servidor en introducir la noción de **proxy** que permite manejar un cache de los documentos encontrados.



**Plexus**

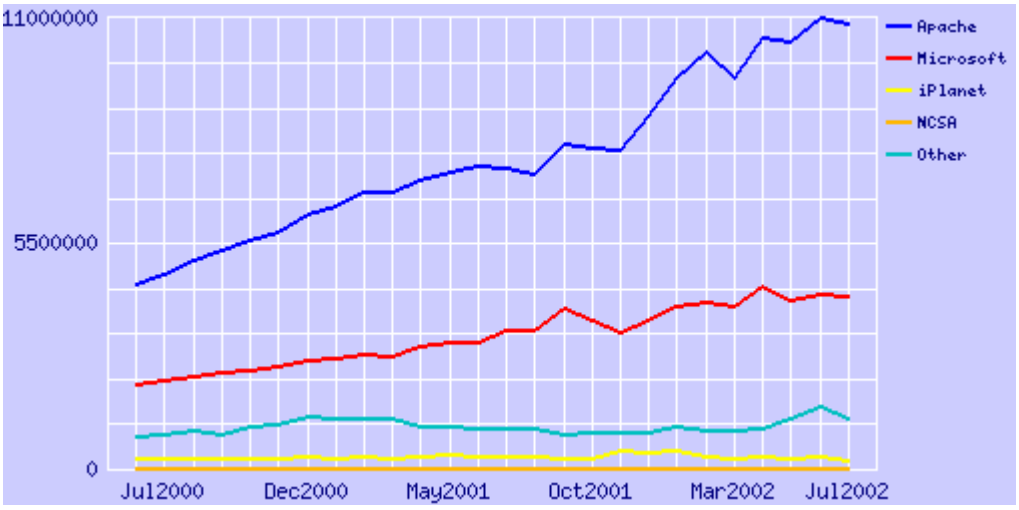
Escrito en Perl.
Ofreciendo mucha flexibilidad de programación, sobretodo ha sido utilizado hace varios años. Pero al ser menos rápido que los otros programas servidor, esta en desuso.

**WN**

Este servidor para estaciones Unix incorpora la posibilidad de efectuar búsquedas de documentos, así como la posibilidad de combinar todo un conjunto de documentos como un solo documento virtual a fin de facilitar la impresión. La búsqueda de documentos puede hacerse sobre la base de los títulos, de palabras-claves, o incluso del contenido completo de los documentos.

En julio de 2002, los servidores más utilizados a través del mundo eran:

## 8.3 Comparativa




(Imagen extraída de *http://www.glug4.linux.org.ar/documentos/charla_marroyo/*)

Situación en julio 2002

**Plataformas de servidores en Internet**

| Plataforma | uso |
|---|---|
| Linux | 35,79% |
| Microsoft | 21,32% |
| Solaris | 20,95% |

Apache, pensado para plataformas Unix, ha estado siempre en continua competición con su adversario IIS, planteado para sistemas Windows de

Microsoft, la diferencia que posiblemente ha marcado la diferencia ha sido la capacidad y el esfuerzo del primero por su implementación en plataformas no nativas.

Aparentemente, la batalla la gana apache, que siendo software libre, funciona en un 70% de los servidores actuales.

Es de destacar que, aunque éste funciona mejor en su plataforma original, es decir, Unix, también tiene un buen rendimiento en el resto de plataformas para las cuales esta implementado.

## 8.4 Funcionamiento

El funcionamiento genérico de un servidor Web se puede resumir en los siguientes pasos:

| | |
|---|---|
| 1. Esperar conexiones TCP al puerto del servidor (el puerto estándar es el 80) | |
| 2. Acepta una conexión | |
| 3. Lee en mensaje de petición de la misma | |
| 4. Interpreta la cabecera recibida | |
| 5. Extrae de ella el path del fichero solicitado | |
| 6. Abre el fichero en su sistema de ficheros | |
| 7. Crea y envía el mensaje de respuesta | |
| | 7.1. Cabecera de respuesta |
| | 7.2. Fichero (lee de disco, escribe al socket) |
| 8. Cierra la conexión | |
| 9. Vuelve al paso 2 | |

## 8.5 Navegadores que se utilizan

Los más conocidos son el Explorer de Microsoft y el Netscape de Netscape Communications Corporation en Estados Unidos y otros países. Tienen capacidades diferentes y es importante cuando se crea una página Web, además de un buen diseño, tener en cuenta la compatibilidad, es decir, programar páginas de modo que las acepte cualquier Navegador.



# 9. HTTP

El Protocolo de Transferencia de HiperTexto, HTTP, 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.
La versión usada actualmente es la 1.1, cuya RFC es 2616

## 9.1 Descripción general del protocolo http

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 (documento HTML, fichero multimedia o aplicación CGI) es conocido por su URL.

## 9.2 Principales características del protocolo http

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 ejemplo, 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. 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.
HTTP se diseñó específicamente para el World Wide Web por lo cual es un



protocolo rápido y sencillo que permite la transferencia de múltiples tipos de información de forma eficiente y rápida. Se puede comparar, por ejemplo, con FTP, que es también un protocolo de transferencia de ficheros, pero tiene un conjunto muy amplio de comandos, y no se integra demasiado bien en las transferencias multimedia [Feit, 1999].

## 9.3. Proceso

1. Un cliente http realiza una petición de un objeto contenido en una URL
2. Se decodifica la URL separando el protocolo, la dirección del servidor, el objeto y el puerto opcional

   http://www.redhat.com/index.html las partes son:
   Http → protocolo
   www.redhat.com → dirección

3. Se establece una conexión TCP, con el servidor remoto, por el puerto 80

4. Se realiza la petición del objeto (pagina web)

5. El servidor envía el objeto requerido mas información, y cierra la conexión6. El proceso anterior se repite tantas veces como sea necesario hasta completar la página

## 9.4. Peticiones

**Método HEAD**Recupera la información de cabecera, pero no envía el cuerpo de identidad
Este método suele usarse para verificar la validez de los enlaces de hipertexto
**Método GET**
Recupera la información especificada por el URL de la petición
**Método POST**
Se emplea para enviar datos al servidor y solicita que éstos sean enviados adecuadamente según recurso de petición
**El método POST está diseñado para encargarse de:**

- Formularios
- Interacción con base de datos
- Envío de noticias de la red

La estructura de una respuesta es:

- versión HTTP + código de estatus de 3 dígitos + descripción textual

-cabecera (conjunto de variables que se incluyen en los mensajes HTTP, para modificar su comportamiento o incluir información de interés

-línea en blanco

-cuerpo del objetoHTTP/1.1 200 OK
Date: Wed, 01 Dec 1999 17:40:18 GMT

Server: Apache/1.3.9 (Unix) (Red Hat/Linux)
Last-Modified: Wed, 01 Dec 1999 17:23:55 GMT
ETag: "db80-5f-384559ab"
Accept-Ranges: bytes
Content-Length: 95
Connection: close
Content-Type: text/html

| Código | Categoría | Descripción |
|---|---|---|
| 1XX | INFORMATIVO | Sin usar, reservado para uso futuro |
| 2XX | ÉXITO | La petición ha sido satisfactoria |
| 3XX | REDIRECCIÓN | La petición requiere otra acción antes de completarse |
| 4XX | ERROR DE CLIENTE | La petición contiene un error de sintaxis y no puede efectuarse |
| 5XX | ERROR DE SERVIDOR | La petición fue válida pero el servidor no puede efectuarla. |

**200 OK** Operación realizada satisfactoriamente.
**301 Moved Permanently**
El objeto al que se accede ha sido movido a otro lugar deforma permanente.

**302 Moved Temporarily**
El objeto al que se accede ha sido movido a otro lugar de forma temporal.
**400 Bad Request**
La petición tiene un error de sintaxis y no es entendida por el servidor.
**403 Forbidden**
Está prohibido el acceso a este recurso. No es posible utilizar una clave para modificar la protección.
**404 Not Found**
La URL solicitada no existe.
**500 Internal Server Error**
El servidor ha tenido un error interno, y no puede continuar con el procesamiento.
**501 Not Implemented**
El servidor no tiene capacidad, por su diseño interno, para llevar a cabo el requerimiento del cliente.



# 10. CGI

## 10.1 CGI. Características

• CGI = “Common Gateway Interface”

• Define una forma de comunicarse con un servidor Web:

- La forma de obtener del servidor Web información sobre las peticiones del cliente Web (Ej: contenido de formularios)

- La forma de darle al servidor Web un nuevo documento que enviar al cliente (Ej.: los bytes de una imagen o el código HTML de una nueva página Web)

• Los comúnmente llamados “CGIs” son simples programas preparados para ejecutarse en un sistema operativo en concreto.

• ¿En qué lenguaje hay que escribirlos? En casi cualquiera: C, C++, Pascal, Fortran, Perl, script de una shell, Python, Tcl/Tk, AppleScript, Visual Basic...

• ¿Qué se debe poder hacer para comunicarse con el servidor Web?:

– Para recibir información del servidor Web debemos: poder leer de la entrada estándar y leer el contenido de las variables de entorno

- Para mandar información al servidor Web debemos: poder escribir por la salida estándar

## 10.2 Tipos de lenguajes a emplear

Tenemos dos tipos de lenguajes para elegir:

**- Lenguajes compilados**

Cuyas características son:

- Hay un código fuente que es compilado en un fichero ejecutable en el lenguaje de la máquina en que vaya a ejecutarse

- Se puede distribuir solo el ejecutable con lo que se mantiene control sobre el código

- Pero hay que compilar el código para la combinación procesador y Sistema .Operativo. en concreto en que se vaya a ejecutar

- Generalmente son programas más pequeños y rápidos, que los programas interpretados



- Se pueden citar entre ellos los lenguajes: C, C++, Pascal...

- Lenguajes interpretados

Cuyas características son:

- El código del programa es interpretado por otro programa. No se compila sino que un programa compilado (el intérprete) ejecuta las instrucciones que indica el “script”

- Siempre que sobre esa plataforma exista el intérprete se puede ejecutar el script sin cambios

- Generalmente son más lentos porque requieren ejecutar primero el programa intérprete y luego éste es más lento ejecutando las instrucciones que si fuera código compilado

- Suelen ser lenguajes más sencillos de programar y depurar

- Se pueden citar entre ellos los lenguajes: Perl, Tcl/Tk, Python, bash...



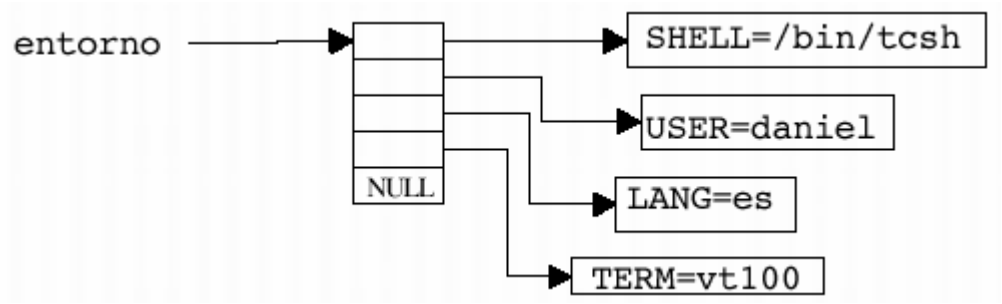
## 10.3 Datos de entrada para un CGI. (Variables de entorno)

• El CGI puede acceder a datos generales sobre el servidor Web, el navegador y la petición. En versiones UNIX se hace mediante variables de entorno:

- Las llamadas " variables de entorno" son un conjunto de cadenas accesibles por el programa.

- En las Shells se puede dar valor y consultar dichas variables. Se heredan al crear nuevos procesos con fork() (lógicamente) y se pueden especificar para un nuevo proceso al ejecutar un programa con execve()

- En C estas cadenas están en un array de cadenas:




- En C se puede conseguir este array de dos formas:

1. Está en el tercer argumento de la función main() : main(int numero_args, char *args_en_linea[], char *entorno[]);

2. Hay una variable global con ese valor con nombre environ que se puede declarar como: extern char **environ;

- Algunas variables de entorno que crea el servidor Web son:

    - REMOTE_ADDR Dirección IP del cliente (el navegador)

    - HTTP_ACCEPT Lista de tipos MIME que acepta el navegador

    - HTTP_USER_AGENT Descripción del navegador (nombre, versión, sistema operativo...)

    - SERVER_PORT Puerto por el que aceptó la conexión el servidor Web

    - SERVER_SOFTWARE Nombre y versión del servidor web

- La forma de modificarlo dependerá del lenguaje en que se escriba el CGI: En C existen unas funciones muy útiles (getenv(), setenv()...)

• El CGI puede acceder a información introducida por el usuario en un formulario. La información puede venir de dos formas diferentes:

- Método GET
- Método POST

Algunas características comunes son:

• Se especifica que se esté empleando un método u otro en la variable de entorno REQUEST_METHOD que valdrá GET o POST y proviene de que en el formulario se especifique el valor del atributo method del tag FORM como uno u otro

• El navegador codifica el contenido del formulario antes de enviarlo. Se llama "URL encoding" (RFC 1738):

- Los diferentes campos del formulario se separan con un ampersand (&)

- Se coloca nombre y valor de cada campo donde el nombre es el valor del atributo name y el valor depende del tipo de elemento

- Los espacios se cambian por el signo +

- Los caracteres " extraños" (generalmente que no están en el US-ACSII o que sea un carácter reservado) aparecen como un signo de porcentaje seguido de un código hexadecimal

Se envía como parte del URL. Ejemplo: http://myserver.org/cgi-in/procesa.cgi?nombre=John+Smith&edad=54

El servidor Web se lo entrega al CGI dentro de la variable QUERY_STRING

**Algunos problemas son:**

- El tamaño máximo de los datos a enviar suele estar limitado

- Se ve el contenido del formulario en el URL

## 10.4 Resumen

- Los CGIs son programa que ejecuta el servidor cuando se le solicita un URI que hace referencia a un CGI

- Pueden obtener información del usuario extrayéndola de los formularios que rellena, a través de variables globales o la entrada estándar

- El resultado del programa por la salida estándar llega al navegador a través del servidor web



• El CGI puede sacar texto, html, imágenes, etc. Debe indicar el tipo (MIME) de lo que saca

- Podemos hacer que el resultado de un CGI se incluya en el contenido de un documento que solicita el navegador (Server Side Includes)

## 11. Requerimientos del servidor web

Se pretende la realización de un servidor web, planteado para una intranet de pequeño o mediano tamaño, que al menos tenga las siguientes funcionalidades:

- Multithread (varios hilos)
- Multiproceso o dirigido por eventos
- Desarrollado en Java
- Funcione en entornos Windows, GNU Linux o heterogéneos
- Soporte CGI

### 11.1 Funcionamiento del servidor web en diversas plataformas

El servidor web ha sido diseñado, originalmente, para Windows (concretamente ha sido probado en Windows XP y Windows-Me) pero también funciona en otras plataformas, con Linux (ha sido probado con Mandrake 10.0).

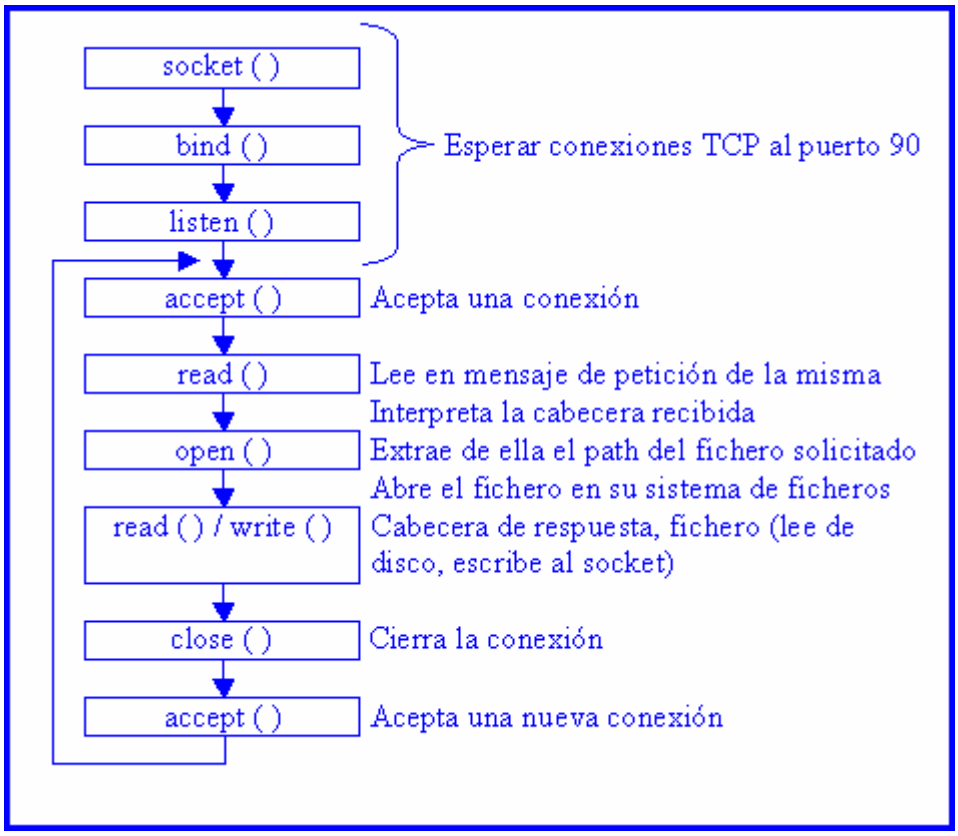
Esto se debe a que su desarrollo ha sido íntegramente en Java, que es multiplataforma.

Cualquier SO que tenga una maquina Java es capaz de ejecutar el servidor web.

Es de destacar y tener en cuenta que las pruebas con CGIs han sido probadas también sobre entornos Microsoft, concretamente he usado una herramienta libre, freePascal (www.freepascal.org).

Si se desease un funcionamiento completo en Linux habría que usar la versión homologa, de freePascal, existente para esta plataforma.

## 12. Organigrama




**Esperar conexiones TCP al puerto 90**
ServerSocket s = new ServerSocket(90);

**Acepta la conexión**
Socket entrante = s.accept();

**Acepta la conexión**
peticionWeb pCliente = new peticionWeb(entrante);
pCliente.start();

**Procesamos la conexión**
BufferedReader in = new BufferedReader(new
InputStreamReader(scliente.getInputStream()));
out = //new PrintStream(new OutputStreamWriter(scliente.getOutputStream())) ;
new PrintStream( scliente.getOutputStream() );

**Cierra la conexión**
in.close();
out.close();

## 13. Partes del servidor

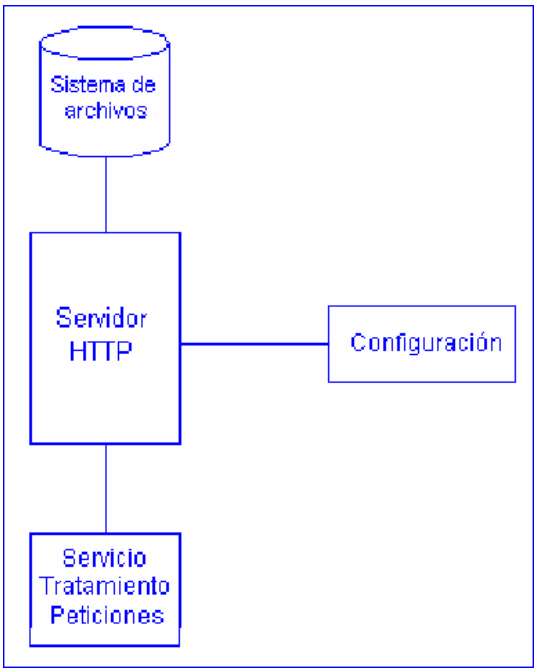
Los módulos que componen el servidor web son tres:

Configuración, en este caso esta integrada dentro del código del servidor pero, en futuras versiones, existirán diversas opciones de configuración que se guardaran en un archivo físico.

sistema de archivos, que gestiona los recursos del servidor, aquí se encuentran los cgis y la base de datos

Tratamiento de peticiones que responde a las diversas solicitudes http del cliente.

La arquitectura se puede ver de manera esquemática en la siguiente figura.




Arquitectura del Servidor http

### 13.1 Módulos

En los siguientes apartados se describe en detalle cada uno de los módulos que componen el servidor desarrollado.

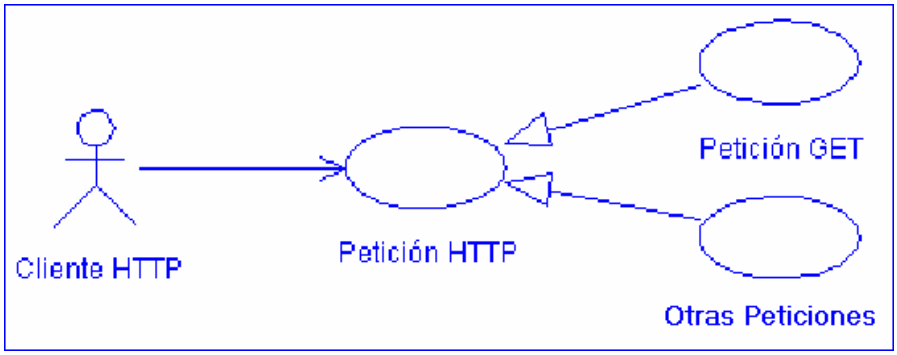



Diagrama de casos de uso del servidor HTTP.

## 13.2. Clases e interfaces

El servidor ha sido diseñado de modo que se minimice la complejidad a la hora de añadir futuras ampliaciones y mejoras. La utilización de patrones de diseño ha sido un factor determinante a la hora de lograrlo, ya que simplifican en extremo los cambios a realizar en el código a la hora de modificarlo.
Las clases y los interfaces existentes se muestran en el diagrama de clases de la figura.

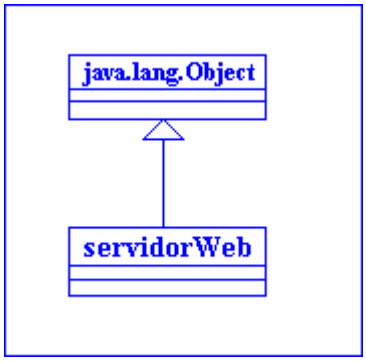
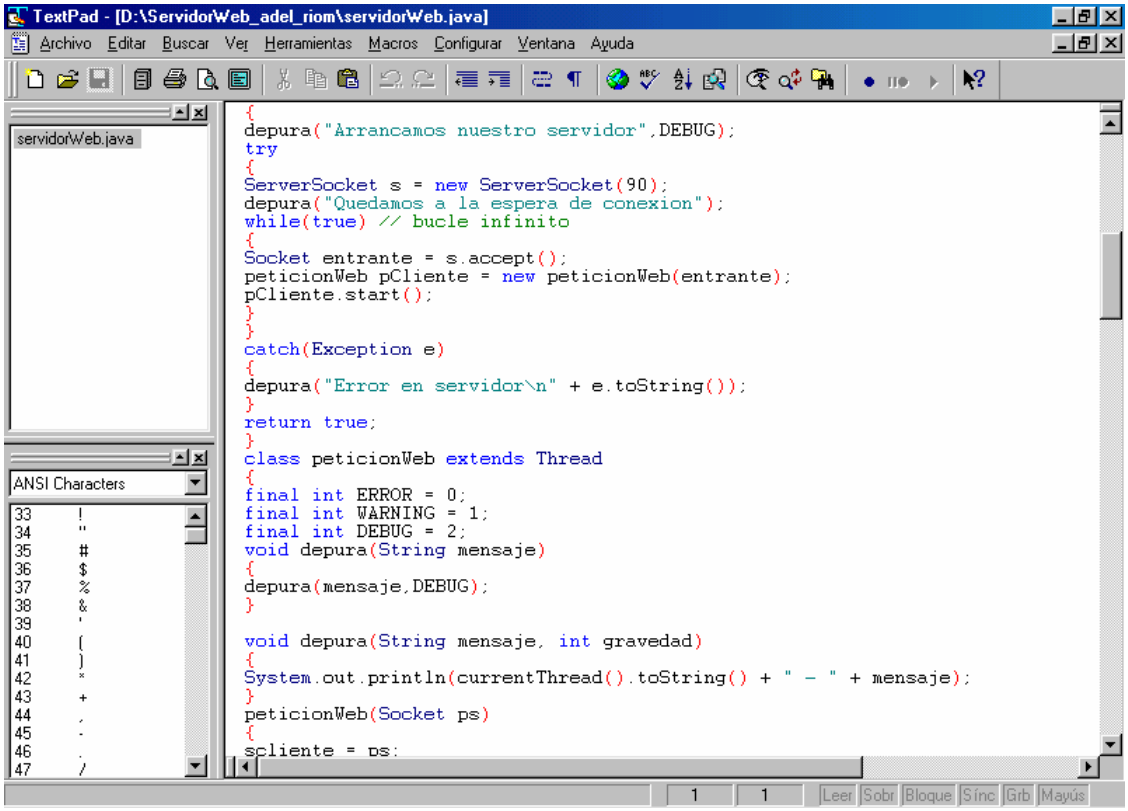



Diagrama de clases servidor HTTP.

## 13.4. Compilación

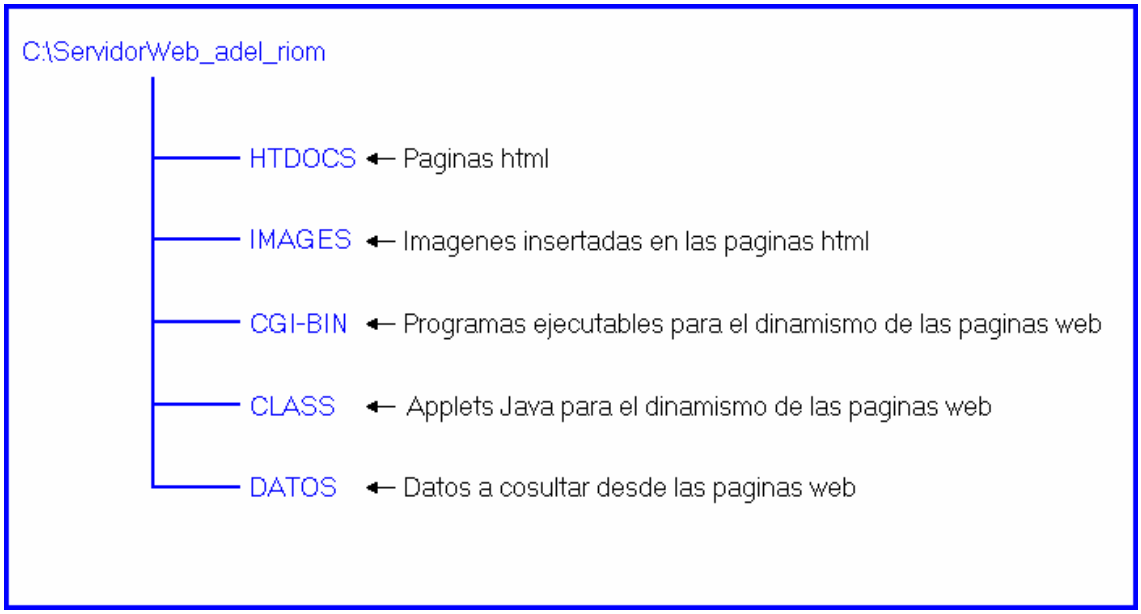
Para compilar el servidor, únicamente hay que escribir, en la consola del directorio **c:\ServidorWeb_adel_riom**, "**javac servidorWeb.java**"
Automáticamente se generaran los archivos **servidorWeb.class** y **servidorWeb$peticionWeb.class.**
Para la compilación y ejecución se ha usado el procesador de textos Textpad.




Detalle de la edición del código fuente del servidor

## 13.5. Configuración

El servidor no necesita ningún tipo de configuración, solo hay que tener en cuenta que trabaja con el puerto 90 (para no interferir con ningún otro servidor, recordemos que por defecto trabajan en el puerto 80) y que debe estar en el directorio C:\ServidorWeb_adel_riom (el cual se considera a todos los efectos el directorio raíz).
Por otra parte es de destacar que los cgis deben encontrarse en el directorio c:\ServidorWeb_adel_riom\CGI-BIN, y las paginas html deben encontrarse en el directorio c:\ServidorWeb_adel_riom\HTDOCS**.**




Detalle del árbol de directorios

Los tipos mime que soporta el servidor son:

- HTM, HTML
- CLASS
- EXE
- GIF
- JPG
- JPEG
- TEXTO

## 14. Tratamiento de peticiones

El tercer y último modulo que compone el servidor WEB va a ser el encargado de llevar a cabo el tratamiento de las peticiones propiamente dichas. Este módulo reconoce a las conexiones entrantes y llevará a cabo las tareas pertinentes para resolver las peticiones HTTP recibidas.

En los siguientes apartados se van a comentar cuáles son los métodos soportados por el servidor, así como las cabeceras de respuesta con las que va a poder responder.

### 14.1. Métodos soportados

Inicialmente sólo se encuentra implementado el método GET.
No se descarta que en futuras versiones el servidor responda a otras cabeceras como HEAD u OPTIONS.

Mediante el método GET se va a poder solicitar cualquier recurso que se encuentre en el sistema de archivos del servidor HTTP. Las peticiones realizadas por el servidor van a realizarse con “tipo de acceso” que será el acceso por defecto.

| Método | Descripción |
|---|---|
| GET | Solicitud de un documento |

Métodos HTTP soportados

### 14.2 Cabeceras de petición

Actualmente no se trata ninguna cabecera, en futuras versiones de la aplicación se podrán realizar un tratamiento adecuado de las mismas. Es la línea de petición lo único que va a tener sentido para el servidor, que analizara el método, el recurso y la versión HTTP del cliente que la realice.

### 14.3 Cabeceras de respuesta

La cabecera de respuesta del servidor HTTP van a ser las mostradas en la siguiente tabla.

| Cabecera | Valor |
|---|---|
| Protocolo | HTTP/1.0 200 Ok |
| Server | Server: TFC-ARM |
| Date | Fecha actual |
| Content-length | Tamaño del documento solicitado |
| Content-type | Tipo mime obtenido de los valores por defecto |

Cabecera de respuesta del servidor http



## 14.4 Errores soportados

Los errores que devuelve el servidor son del tipo 2XX, 4XX y 5XX, ya que no existen respuestas parciales ni redirecciones.

Los códigos devueltos son los siguientes:.

| Resultado | Código | Descripción |
|---|---|---|
| HTTP_OK | 200 | La petición es correcta, se muestra el resultado |
| HTTP_BAD_REQUEST | 400 | La petición es incorrecta |
| HTTP_FORBIDDEN | 403 | No se tiene acceso al recurso solicitado |
| HTTP_NOT_FOUND | 404 | El recurso solicitado no existe |
| HTTP_BAD_METHOD | 405 | El método empleado no existe |
| HTTP_INTERNAL_ERROR | 500 | Se ha producido algún error interno en el servidor |
| HTTP_NOT_IMPLEMENTED | 501 | El método especificado no se encuentra implementado |
| HTTP_UNAVAILABLE | 503 | No es posible atender la solicitud debido a sobrecarga del sistema. Si se repite la solicitud más adelante es posible que pueda ser atendida |
| HTTP_VERSION | 505 | El servidor no entiende la versión HTTP del cliente |

Códigos de respuesta del servidor http

En futuras versiones del servidor cabe la posibilidad de contemplar un mayor rango de errores.

Inicialmente se contemplan los errores relativos a comandos incorrectos, fallos del servidor así como accesos ilegales a recursos del sistema (inexistencia de un documento o acceso no permitido a un recurso).

## 14.5 Ejemplo de funcionamiento

En este apartado se muestra un sencillo ejemplo del tratamiento de una petición HTTP.

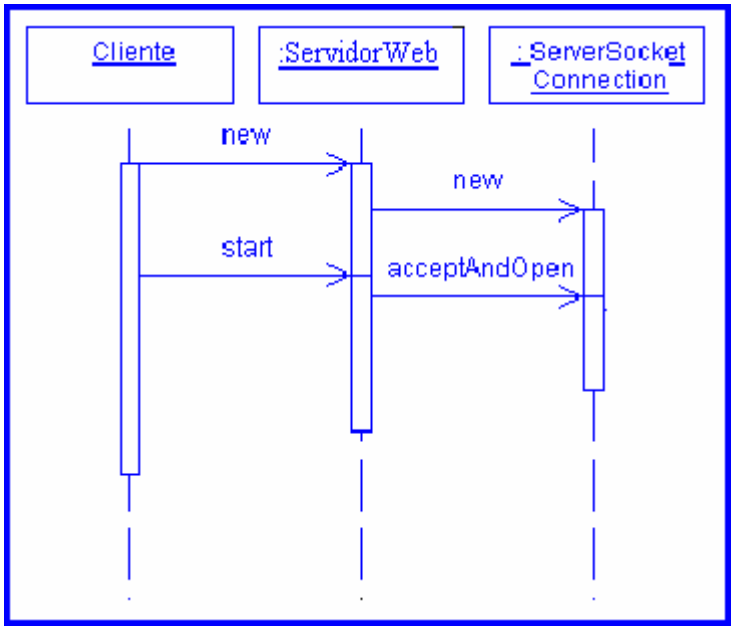
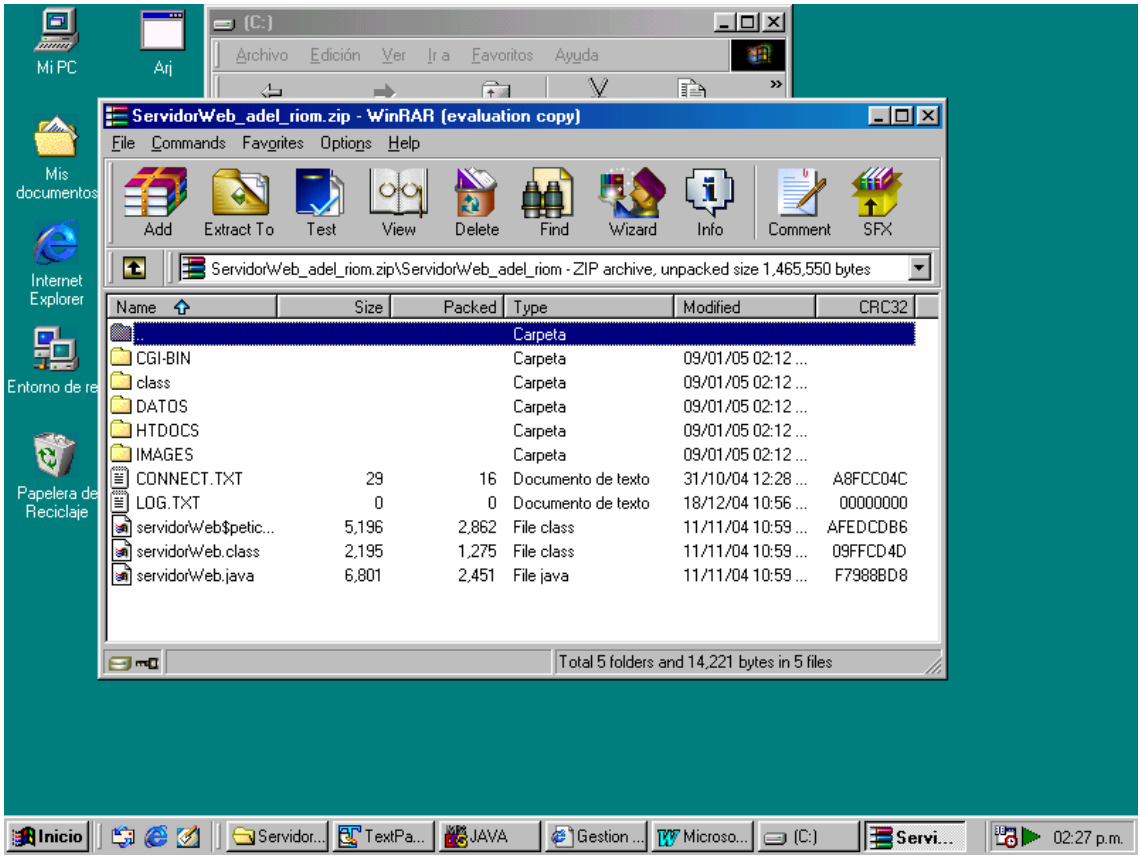



Diagrama de secuencia del servidor http

## 15. Instalación

El formato de distribución de la aplicación va a ser un archivo zip, que únicamente será necesario descomprimir en la unidad C

Dentro del archivo zip se van a incluir no sólo las clases necesarias, si no aquellos recursos (archivos HTML e imágenes en este caso) que va a incluir el servidor HTTP en su sistema de archivos.



Detalle de la instalación

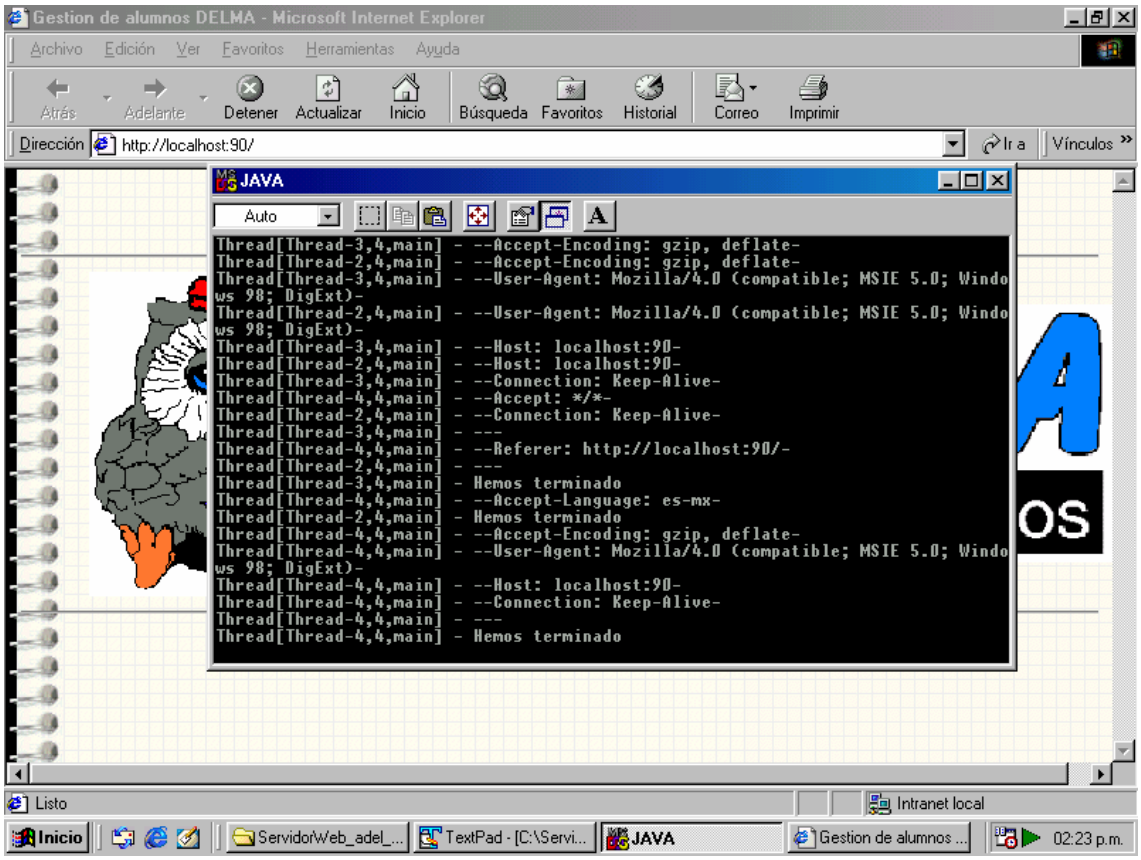
### 15.1 Inclusión de recursos en el servidor

En primer lugar y dentro de la aplicación se han incluido una serie de imágenes y archivos HTML que constituirán los recursos que el servidor va a poder proporcionar inicialmente.

También está incluida una pequeña base de datos, que es necesario conectar al ODBC de Windows para comprobar su funcionamiento.

## 16. Ejecución

Para la ejecución del programa únicamente hay que escribir en la consola **"java c:\ServidorWeb_adel_riom\servidorWeb"**




Detalle de la ejecución del servidor

## 17. Métodos y funciones empleados

**Depura**
Sintaxis: void depura(String mensaje)
Descripción: Función para centralizar los mensajes de depuración

**Main**
Sintaxis: Public static void main(String [] array)
Descripción: punto de entrada al programa

**servidorWeb**
Sintaxis: servidorWeb(String [] param)
Descripción: constructor que interpreta los parámetros pasados

**procesaParametros**
Sintaxis: boolean procesaParametros
Descripción: parsea el fichero de entrada y establece las variables de clase

**arranca**
Sintaxis: Boolean Arranca
Descripción: crea el socket y procesa la conexión

**tipoMime**
Sintaxis: public String tipoMime( String fich )
Descripción: devuelve el tipo mime correspondiente



## 18. Juego de pruebas

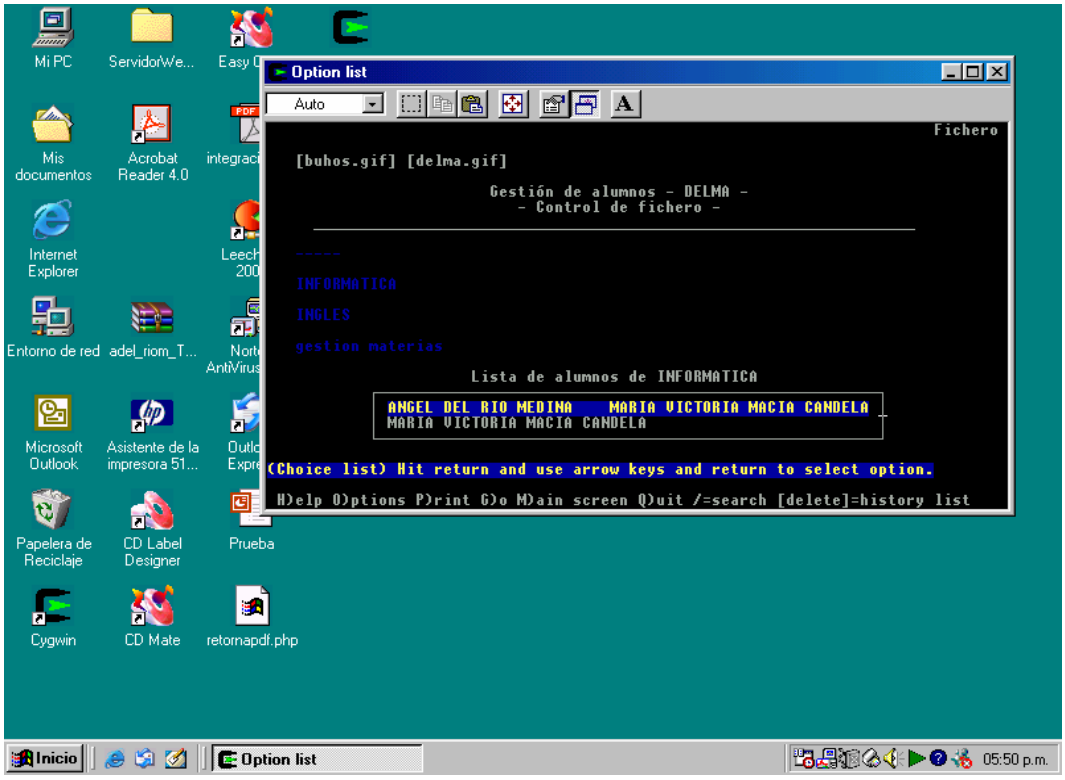
Para el juego de pruebas se ha elegido la implementación de parte de una aplicación para la gestión de un pequeño centro de estudios. En concreto se va a implementar la gestión de los alumnos y las materias.
Esta pequeña aplicación, desarrollada en Pascal, también ha servido para comprobar que el servidor funciona también dentro de una intranet. Se ha probado tanto con clientes con sistema operativo Windows, como Linux, obteniendo en ambos casos un resultado satisfactorio.

### 18.1. Casos de uso

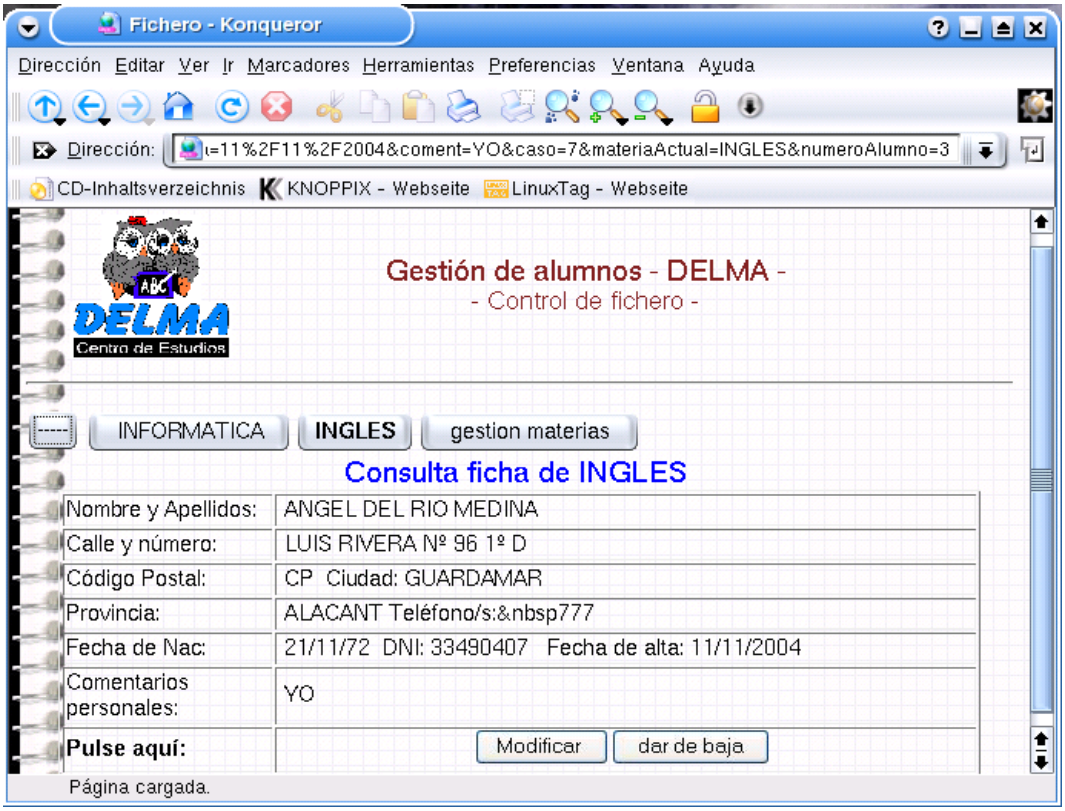
Caso:

- 0,1,2:mostrarListaAlumnos();
- 3:nombreAlumno(Actual);
- 4: nuevoAlumno(contenidoDato('nombre'),Actual) o ConsultaAlumno(contenidoDato('nombre'),Actual); según exista o no el alumno
- 5:consultaAlumno(contenidoDato('nombre'),Actual);
- 6:modificaAlumno(numeroAlumno(),materiaActual());
- 7:actualizaDatosModificados();
- 8:baja(Actual,contenidoDato('numeroAlumno'));
- 9:bajaConfirmada();
- 10:mostrarListaMaterias();
- 11:consultaMateria(contenidoDato('nombre'));
- 12:formularioMateria(materiaActual(),contenidoDato('descripcion'),contenidoDato('precio'),contenidoDato('comentarios'));
- 13:introduceMateria();
- 14:formularioMateria(materiaActual(),'','','');
- 15:grabaMateria();

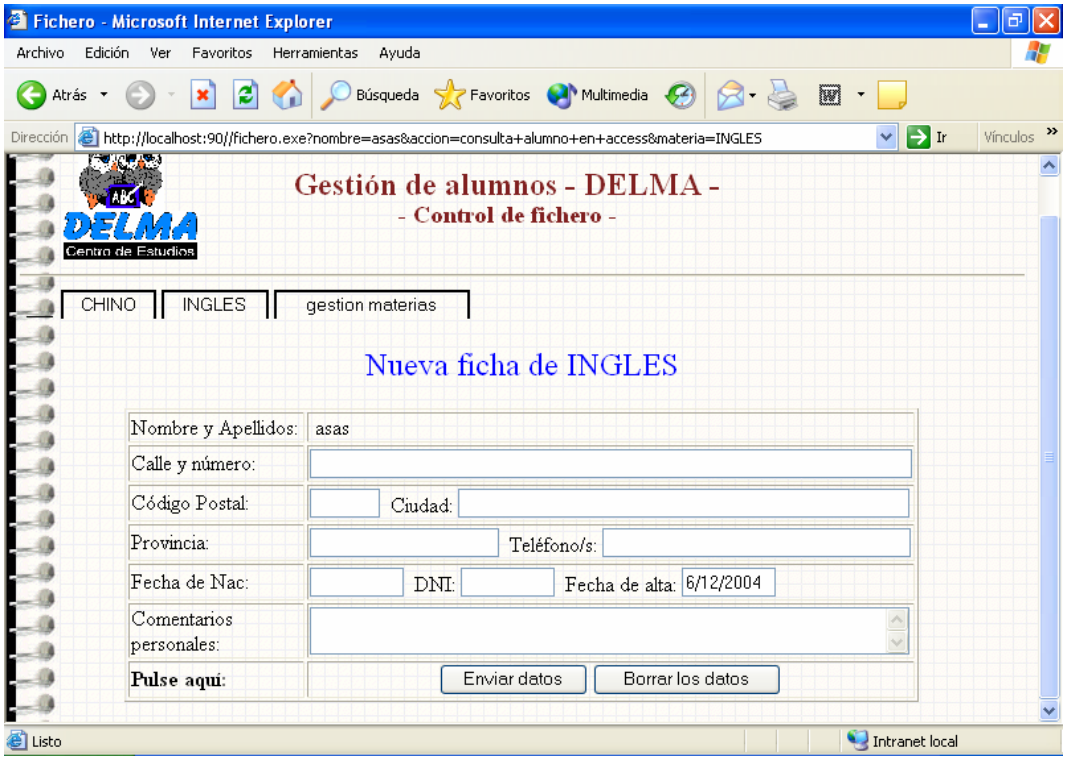
#### 18.1.1. Ejemplos de casos de uso



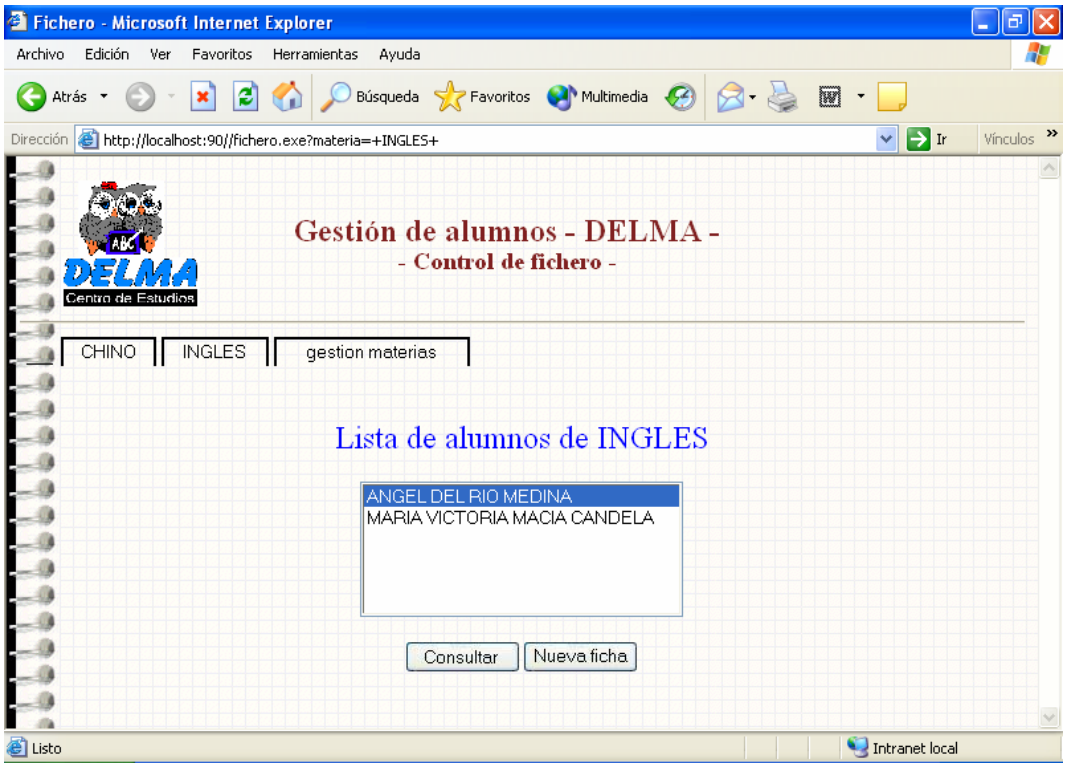

Caso 0, el cliente es Lynx de Cygwin ejecutándose en Windows 98




Caso 5, el cliente es Konqueror en Knoppix, una distribución de Linux




caso 4: el cliente es Internet Explorer en Windows XP




caso 0: el cliente es Internet Explorer en Windows XP

## 19. Conclusiones y futuros trabajos a desarrollar

He tenido la oportunidad de profundizar en el funcionamiento del protocolo http, e implementar en java una pequeña aplicación, que trabajando con este protocolo, es capaz de servir diferentes tipos de archivos.

Para completar este pequeño servidor web, he añadido la funcionalidad CGI, con lo cual el programa, aunque rudimentario, está completo.

Me ha resultado muy gratificante, ver como una pequeña aplicación, enlazada a una base de datos, funcionaba correctamente, tanto de forma local como en una intranet.

Las pruebas con CGIs están hechas en el lenguaje de programación Pascal, pero sería posible usar cualquier otro, C, Cobol, incluso alguno interpretado como Perl.

Como futuros trabajos, hay mucho por hacer, por nombrar algunas cosas:

Implementar algún tipo de mecanismo que permita ejecutar el mismo CGI, sin que se pierdan las variables de entorno, de forma simultánea en dos o más clientes de la red (ahora mismo sólo se puede ejecutar, sin peligro, un solo cgi)

Ofrecer la posibilidad de ejecutar servlets, ahora mismo las únicas formas de ofrece interactividad, son con cgis o applets (los cuales también he probado, siendo correcto su funcionamiento).

Analizar y optimizar el tiempo de respuesta y ejecución del servidor.

Conectar al servidor a Internet, con el fin de hacerlo funcionar, también, como servidor a nivel externo.

Tener en cuenta mecanismos de protección para evitar el uso indebido de la información del servidor por personas no autorizadas.

Documentar y publicar en Internet este trabajo con el fin de que sirva a otros.

Estos podrían ser algunos de los trabajos futuros que se podrían realizar teniendo en cuenta este TFC y tomándolo como base.



## 20. Bibliografía y fuentes de información

### Enlaces

http://www.htmlpoint.com/apache/09a.html

http://www.glug4.linux.org.ar/documentos/charla_marroyo/

http://www.adictosaltrabajo.com

http://barrapunto.com/article.pl?sid=03/12/04/0936237&mode=thread

http://www.htmlpoint.com/iis/

http://www.ilustrados.com

www.freepascal.org

### Documentos

http://congreso.javahispano.org/files/doc/j2me/Servidor_HTTP_para_J2EE.pdf

Proyecto fin de carrera
DISEÑO E IMPLEMENTACION DE UN SERVIDOR http Y MECANISMOS DE SERIALIZACION EN J2ME
Autor: Guillermo Diez-Andino Sancho (gdandino@it.uc3m.es)
Tutora: Celeste Campo Vázquez (celeste@it.uc3m.es)
Fecha: 23 de septiembre de 2002

### Libros

Diccionario de programación en Pascal
Autor: Luis Navarro Larred
Editorial: Inforbooks

Java 2. Curso de programación
Editorial: Rama
Autor: Fco. Javier Cevallos

Cómo se hace con Java
Autores: Madhu Siddalingaiah, Stephen D. Lockwood

Utilizando LINUX
Autores: Tacket & Gunter
Editorial: Prentice Hall