Arquitectura de un Captive Portal: Toc^2

Realizado por: Jorge Hortelano Otero
Dirigido por: Pietro Manzoni

UNIVERSIDAD
POLITECNICA
DE VALENCIA

15 de febrero de 2006

Agradecimientos.

Como ya va siendo tradición en todos los proyectos finales de carrera, se suele escribir una sección de agradecimientos al principio del mismo. Posiblemente esto ocurra ya que para los estudiantes, durante unos meses, nuestra vida gira exclusivamente en torno a la creación de este documento. Cuando uno llega al final de la redacción del proyecto, se da cuenta que existe alrededor nuestro una gran cantidad de gente que ha estado preocupándose y mostrando su apoyo en la elaboración del mismo. Para ellos, se dedican las siguientes líneas:

Primero, a mis padres, que aunque creo que nunca han llegado a saber realmente lo que he estado haciendo durante todas estas horas, han confiado en mi y me han dejado hacer sin dudar un solo instante.

A mi director del proyecto, Pietro, pues sin su interés por este trabajo, este no hubiera sido nunca lo que ha llegado a ser. También debo un agradecimiento a Juan Carlos Cano y a Carlos Calafate, pues mientras que algunas personas solamente han tenido un único director de proyecto yo he tenido prácticamente a tres.

A mis amigos, pues solamente ellos saben cuantas horas me he pasado hablando de un tema que probablemente solamente yo comprendo. A veces hay que tener mucha paciencia para soportar esto y ellos la han tenido conmigo.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción.

Este documento es la consecuencia del trabajo invertido por el autor en la realización de su Proyecto Final de Carrera. Este proyecto trata sobre la elaboración y diseño de un *Captive Portal*. Un *Captive Portal* o Portales Cautivos, como lo traducen algunas personas [15], es un portal que fuerza a un navegador de Internet a ver e interactuar con una página web especial, usualmente con motivos de autentificación, en vez de presentar la página web solicitada de forma normal.

Cuando un usuario potencial accede a una red con un *Captive Portal* [12], una página web aparece y solicita ciertas acciones antes de permitir el acceso a Internet. El ejemplo más simple de *Captive Portal* fuerza al usuario como mínimo a leer una política de uso del mismo e indicar la aceptación de la misma. Presumiblemente, esto puede ayudar al proveedor a evitar responsabilidades si un usuario realiza algún acto criminal. En otros proveedores, lo que se muestra es el logotipo del mismo, y el usuario tiene que acceder a través de este a Internet. Hoy en día, en los más utilizados, se requiere para entrar que el usuario introduzca una identificación y una contraseña preestablecidos. La existencia de una cuenta de usuario puede disuadir a algunas personas la utilización de estas redes para realizar actos criminales.

Otra ventaja es que muchos de estos *Captive Portals* tienen además antivirus y *firewalls* que protegen a los usuarios que acceden a través de ellos a Internet.

Incluso cuando el más simple de los *Captive Portals* es usado en una red de acceso libre, ciertas personas pueden conectarse repetidamente, usando la red para la descarga de música, vídeos y otros archivos de gran tamaño. Esta actividad, denominada por los expertos "*bandwidth hogging*" puede ser minimizado mediante una programación adicional al crear el *Captive Portal*. Esta programación puede controlar la velocidad a la que se realiza la descarga de estos archivos, permite limitar el tamaño máximo de los archivos que se pueden descargar, restringir el número de descargas que pueden existir en una sola sesión o bloquear sitios web que comúnmente se utilizan para descargar estos archivos. A esto se le llama "*bandwidth throttling*" o "*traffic shaping*". Existen ya diferentes implementaciones de este sistema en el mercado, muchos de ellos de código libre que pueden ser descargados y usados gratuitamente.

Aparte de los motivos de control y seguridad citados anteriormente, existen también motivos económicos para su implantación. Tushar Sachdev [11] explica el surgimiento de esta tecnología como una necesidad económica, ya que:

1. Reduce el coste al realizar un cambio tecnológico en una comunidad.
2. Las aplicaciones se pueden desplegar en un ámbito más amplio, reduciendo el coste por usuarios.
3. La unificación de estructuras de autentificación, seguridad web, etc. reduce el coste de la infraestructura.
4. El uso de portales ya desarrollados permite a las nuevas empresas adoptarlos rápidamente sin realizar cambios de código, permitiendo su rápido desarrollo a un coste muy bajo.

Estos motivos hacen que el uso de los *Captive Portals* se extienda cada vez más hoy en día. Por ejemplo, el Centro Experimental de Comunicaciones Inalámbricas de la universidad Rey Juan Carlos [4], señala que raro es ya encontrarse una empresa que no tenga parte de su red local construida con tecnología inalámbrica: gran parte de las universidades públicas y privadas dan acceso Wi-Fi a sus alumnos y empleados, aeropuertos, hospitales, cadenas hoteleras, empresas de restauración,... la posibilidad de estar siempre conectado es, cada vez más, una realidad. Una forma de controlar esta conexión, es mediante el uso de estos *Captive Portals*.

El interés de este trabajo no consiste tanto en presentar una herramienta que compita con las ya existentes en el mundo, sino en el desarrollo de una estructura que permita integrar esta tecnología con otro servicio ya existente de una forma completa.

A esta arquitectura lo llamaremos a partir de ahora Toc^2 o simplemente TocToc (que representa el sonido *toc-toc,* que se produce al llamar a una puerta con el puño).

Capítulo 2

Objetivos.

Este proyecto, lo que pretende es el diseño y la implementación de una arquitectura para el desarrollo y despliegue de Portales Cautivos basados en software libre. Se busca que sea totalmente integrable con un portal web que ofrezca cualquier otro servicio. Intentar crear una herramienta capaz de realizar lo mismo que cualquier *Captive Portal* del mercado, pero que no tenga ningún problema a la hora de fusionarse con otros servicios que puede necesitar un portal de acceso en concreto. Para ello, en esta memoria se describirá el cómo se ha diseñado para conseguir este fin: primero una estructura general en la que se explica la idea del portal, el funcionamiento y los lenguajes que se utilizarán para llevarlos a cabo; seguido de una implementación más técnica del mismo, con los detalles y lenguajes utilizados (capítulos 3 y 4). Se presentarán también diversos resultados que se han obtenido en la experimentación con el mismo (capítulo 7), mostrando el comportamiento con diversos clientes que representan casos extremos. De esta forma se probará el funcionamiento correcto del sistema ante diversas situaciones mas o menos adversas. En la tercera parte se pretende mostrar varios ejemplos prácticos que representen la futura utilización del portal (capítulo 8). Una vez se sabe que el portal funciona, resulta interesante ver que salida puede tener en el mercado actual y su utilidad. Estos ejemplos son los sistema Dulenduè y RuralNet. El primero de ellos utiliza las técnicas de localización inalámbricas para ofrecer un servicio dependiente de la localización del usuario El segundo es lo que se denomina *Internet Service Provider* (ISP) ideado para entornos rurales en los que difícilmente se puede instalar otra estructura. Por último, se presenta un conjunto de mejoras posibles a realizar en el futuro (capítulos 9).

Capítulo 3

Arquitectura propuesta.

3.1. Infraestructura necesaria.

La infraestructura mínima para poner en marcha el sistema, contiene sin duda un equipo que haga las funciones de servidor. Este equipo tiene que tener acceso a Internet con banda ancha por un lado y por otro, debe de poder generar una red *wireless*, ya sea por tarjeta propia o mediante un punto de acceso conectado al equipo. Este equipo será el encargado de centralizar toda la gestión de usuarios y permitir la salida a Internet.

Por otro lado, el servidor debe de estar conectado a una estructura *wireless*. Esta estructura debe tener un alcance lo suficientemente elevado para que todo el sistema tenga sentido, como por ejemplo un recinto o una localidad. Esto se puede conseguir mediante la colocación de distintos nodos a lo largo del área a cubrir y la conexión entre ellos mediante una red convencional o mediante otra red *wireless*.

Los clientes, solamente necesitan disponer de una conexión *wireless*. Por cliente entendemos cualquier dispositivo que pueda visualizar un navegador, ya sea un ordenador portátil, una PDA o un teléfono móvil con estas capacidades.

Todo esto se observa en la figura 3.1, en la que aparecen recuadrados las distintas partes de la arquitectura antes comentadas.

Los programas que tiene que tener este equipo instalados, viene detallado en el punto 4.1 que se presenta más adelante.

3.2. Formas de conexión entre los equipos.

La forma de conectar el cliente al servidor tiene que ser mediante la creación de una red inalámbrica sin protección alguna. No puede tener encriptación *WEP* ni *WAP* ya que eso implicaría el conocimiento de la clave de red y por tanto no sería público el acceso.

El tipo de red resulta indiferente y dependerá del administrador decidir lo que desea según los medios disponibles. En las pruebas utilizadas se ha probado el funcionamiento en una red en modo *Ad-Hoc* y en otra mediante un punto de acceso obteniendo datos positivos en ambos casos. Sin embargo, habrá que tener en cuenta que esto es un ejemplo de prueba y que las dimensiones de la red destino probablemente sean mucho mayores por lo que resulten necesarios

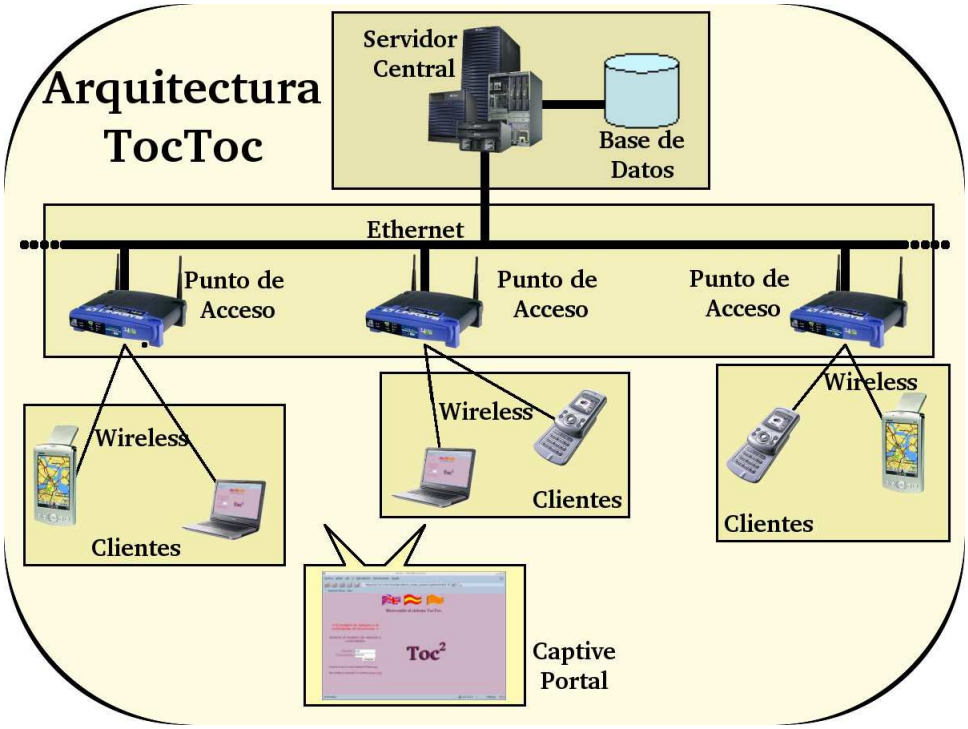

Figura 3.1: Arquitectura general.

una gran cantidad de puntos de acceso para dar cobertura a la misma. Para ello habrá que tener en cuenta la interacción entre ellos y los problemas derivados del mismo.

3.3. Lenguajes utilizados.

Los lenguajes utilizados para el desarrollo del programa han sido varios dependiendo del problema que se quiera abordar. Algunas tecnologías son mejores que otras para tratar ciertos problemas y otras son, sin embargo, mucho mejores para obtener una compatibilidad entre sistemas. Por ello, se ha decidido el uso de los siguientes lenguajes:

1. HTML: el lenguaje básico para la creación de una página web es, por excelencia, el HTML. Con este lenguaje se ha desarrollado el aspecto básico del sistema que es el portal que recibe a los clientes. Permite generar la estructura de cada una de las páginas por las que se va moviendo el cliente.

2. CSS: el lenguaje de estilos en cascada, permite dar un aspecto homogéneo al portal, de manera que se controle todos los aspectos de formato de las fuentes, tamaños de texto y títulos de una forma estándar a todas las páginas.

3. PHP: este lenguaje permite al portal dar a las páginas una personalización

para cada cliente que no hubiera sido posible de otra forma. Con PHP se abordan todas las comunicaciones con la base de datos, así como la genración de la sesión del cliente al entrat en el sistema. Permite también adaptar la página a los permisos que tenga este cliente en la misma. Por ejemplo, permite que solamente los administradores del sistema tengan acceso a las zonas de administración.

4. SQL: es la sintaxis de la base de datos que se encarga de mantener el sistema de usuarios y servicios a lo largo del tiempo, permitiendo consultas, actualizaciones y borrados mediante la interacción con PHP.

5. Javascript: este lenguaje permite solucionar algunos problemas sencillos en la máquina del cliente, liberando al servidor del sistema de algunas tareas que no tiene por que realizar él necesariamente. En este caso, se utiliza para realizar una primera comprobación de los datos, que debe introducir el cliente en los formularios que aparecen en el sistema antes de ser enviados al servidor. También sirve, por citar otro ejemplo, para controlar el tiempo de conexión de un cliente a Internet, ya que la hora del mismo es local y por tanto dependiente de la máquina en que se ejecuta.

6. XML: usar este lenguaje de etiquetas en algunos aspectos del sistema, permite la compatibilidad futura con otros entornos. En el sistema principalmente se ha utilizado XML para la opción de multilingüismo[1]. De esta forma se permite el desarrollo futura de una aplicación que permita la actualización o ampliación del multilingüismo de una forma sencilla sin conocimientos de programación.

7. XSL: al igual que lo es el CSS para el HTML, el XSL permite tratar el formato de las páginas presentadas en lenguaje XML.

8. DHTML: el HTML dinámico permite algunas mejoras gráficas que serían demasiado pesadas de realizar mediante Javascript o HTML únicamente. En la este caso, este lenguaje se usa exclusivamente para desarrollar el menú de navegación que guía al cliente por toda la páginas.

9. Actionscript (Flash): La presentación principal del sistema[2]tiene un aspecto gráfico más elaborado. Aunque esto no es esencial en el sistema, es común utilizar este lenguaje para dar un toque artístico en el diseño de la página.

10. Shell de Linux: toda la parte de control de acceso a la red y la gestión del ancho de banda, se realiza mediante las aplicaciones TC e IPtables integradas en el kernel del sistema Linux. La configuración del mismo se realiza mediante la línea de comandos o mediante la generación de un *script* que realice los pasos necesarios. Esto permite la automatización posterior mediante el *cron* de Linux o la ejecución de los mismos mediante consola.

Algunos criterios extra para seleccionar tanta diversidad de lenguajes, es sin duda, la ejecución del lenguaje en la maquina cliente o en la máquina servidor. Por ello, como se observa en la figura 3.2 se intenta que la mayoría de los

[1]Para mas información sobre el mismo, dirigirse a la sección 4.10.
[2]Se puede ver esto en los ejemplos de uso que aparecen en el capítulo 8.

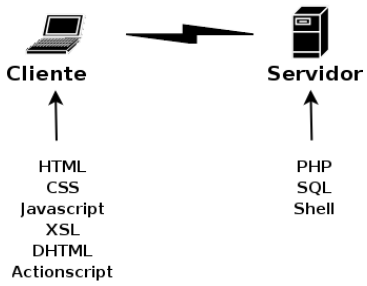

Figura 3.2: Ejecución de los distintos lenguajes utilizados.

lenguajes se ejecuten en la máquina cliente ya que en caso contrario el servidor acabaría saturándose rápidamente.

Capítulo 4

Implementación.

La arquitectura TocToc pretende controlar el acceso a Internet a través de una computadora que hace la función de *Gateway*. Para ello, debe de tener la capacidad de poder controlar que clientes salen a Internet y cómo lo hacen. En este capítulo se pretende dar a conocer como se consigue realizar esta tarea. La explicación se divide en varios apartados:

- Una introducción a las herramientas que se utilizan. Presentación de los programas externos que se utilizan y de lo que se encarga cada uno de ellos.
- Cómo se consigue el control de acceso a la red externa por parte del sistema. Permitir al sistema controlar que equipos de la red local pueden salir al exterior. Esto incluye:
  - El acceso al portal TocToc. Cómo hacer que un cliente que desconoce la existencia del mismo llegue hasta él de forma involuntaria. Si no, resulta imposible que el cliente pueda darse de alta en el sistema y conseguir el acceso al exterior.
  - El control de acceso a Internet, limitándolo solo a los clientes que lo han contratado e impedir el acceso al resto.
  - El acceso a algunos servidores a todos los usuarios, independientemente de si han contratado o no conexión a Internet. Ya sea por que parte del sistema está fuera de la máquina local o por que se pretende dar acceso información externa, se puede permitir el acceso general a algunas máquinas concretas.
- El control de los clientes: cómo darlos de alta, guardar sus datos y permitir el acceso a Internet a cada uno de ellos por separado.
- Registro de los acceso al portal. Es conveniente que una arquitectura de este tipo tenga un control de que cliente accede a Internet a través de él y cuando lo hace. Importante tanto por motivos de seguridad, como por motivos de tarificación.
- Permitir el uso de varios idiomas en esta arquitectura. La aplicación puede ser utilizada por personas de distintas nacionalidades y puede ser necesario

una traducción transparente para cada uno de ellos. A esto se le denomina multilingüismo.

- Como exportar esta arquitectura a otros sistemas. Esto no es más que una arquitectura básica que pretende ser la base de otros sistemas mucho más complejos.

Todo esto se explica a continuación.

4.1. Herramientas utilizadas.

Para la realización del proyecto, debido al carácter innegable de estilo web, se han utilizado distintas herramientas para aplicación del mismo. En el capítulo anterior, se han enseñado los distintos lenguajes necesarios para la generación del sistema, ahora falta decidir que herramientas se adoptarán para la utilización del mismo. A continuación se presenta una enumeración de las mismas y se detalla su función:

1. Apache: el sistema tiene una interfaz desarrollada vía web que es lo que se le muestra al cliente cuando entra. El servidor Apache permite servir páginas web de forma fiable y segura mediante SSL, motivo por el cual fue elegido para la implementación del sistema. La versión de Apache elegida al desarrollar el sistema es la 2.0.53.

2. OpenSSL: para dar seguridad al sistema y a los datos transmitidos, el sistema envía las páginas cifradas mediante el protocolo SSL. La versión con la que se ha probado el sistema es la 0.9.7e.

3. PHP: este preprocesador de hipertexto permite el desarrollo de páginas web dinámicas, la interacción con bases de datos y el uso de comandos UNIX. Todo ello indispensable para el funcionamiento del sistema. La versión de PHP elegida es la 5.0.3. Se conoce de la existencia de incompatibilidades con versiones inferiores a la 4.1.

4. MySQL: el sistema necesita un soporte para el almacenamiento de los usuarios dados de alta, así como de las distintas conexiones permitidas a los clientes. Para ello se utiliza una base de datos. La elección de esta base de datos viene determinado por su carácter gratuito. La versión mínima para el correcto funcionamiento del sistema tiene que ser la 4.1.10.

5. IPtables: esta herramienta que viene vinculado al kernel de Linux es un sistema de *firewall* que permite seleccionar que paquetes pasan por la red según el destino o el origen. Una herramienta fundamental para impedir el acceso a Internet a los clientes no autorizados. Este programa viene con cualquier kernel superior al 2.4. Para más información sobre el uso del mismo, se puede consultar el manual [7].

6. TC: es otra herramienta integrada en el kernel de Linux encargada de controlar el tráfico de red. Esta herramienta se usa en el sistema para controlar la velocidad de descarga o subida en la red según los permisos a los que tengan acceso. Al igual que IPtables, este programa aparece con las versiones del kernel actuales.

7. DHCP: tener este servidor instalado es importante, ya que todos los clientes que se conecten a la red desconocerán las características de la misma. El servidor DHCP se encargará de darles a cada uno una *IP* e indicarles los demás datos como la Puerta de Enlace y el DNS.

4.1.1. Relación entre los programas utilizados.

Una vez ya están elegidas todos los lenguajes necesarios para realizar lo que se pretende, hay que ver como se consigue relacionar todo este conjunto de elementos para que el proyecto cobre algo de sentido. La figura 4.1 muestra gráficamente como funciona esto. Como se puede ver, por un lado el servidor Apache presenta directamente los datos estáticos del sistema, mientras que PHP se encarga de interactuar con todos los programas que dependen su ejecución del estado del sistema. La flecha simple indica lectura de datos, mientras que la flecha doble indica interacción entre programas.

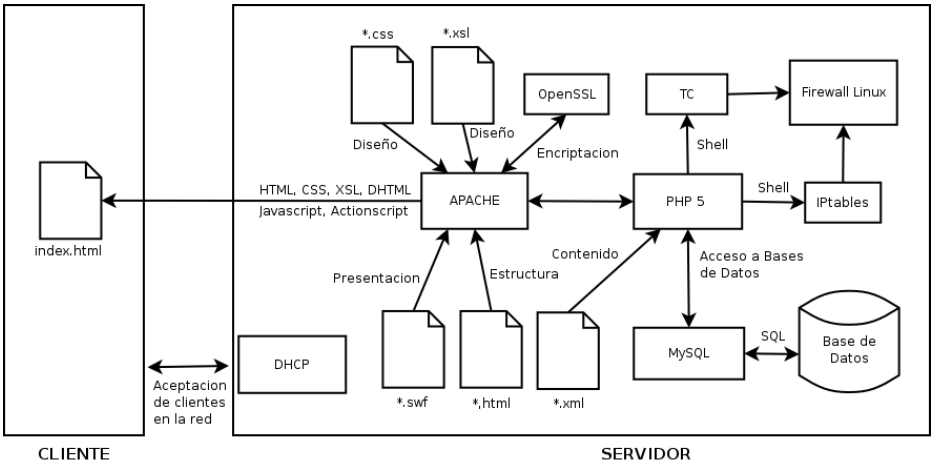

Figura 4.1: Relación entre programas, lenguajes y máquinas.

Como se puede observar, el servidor genera un conjunto de datos que envía al cliente como una página web que recibe y ejecuta el cliente. De esta forma se consigue la compatibilidad para cualquier equipo que tenga una navegador.

4.2. Captura de peticiones de un cliente no reconocido.

La arquitectura debe de capturar todas las peticiones hacia servidores externos que pasen a través de la máquina que contiene la arquitectura TocToc y no pertenezcan a clientes que ya tengan permiso de salida a Internet. Para ello, se activa el *script* que inicializa el servidor. Concretamente, de ese *script*, lo que realiza este proceso son las siguientes líneas:

Algorithm 1 Captura de clientes.

```
    #Variables.
IPT=/sbin/iptables
PWLAN="192.168.1"
TOCTOC="10"
DHCPmin="195" #Para el cliente 192.168.1.195
DHCPmax="205" #Hasta el cliente 192.168.1.205

#Preparo todos los clientes que captura el servidor DHCP.
for ((i = $DHCPmin; i <= $DHCPmax; i++))
do
$IPT -t nat -A PREROUTING -p tcp -s $PWLAN.$i -j DNAT
–to-destination $PWLAN.$TOCTOC
done
```

Como se puede observar en el código, lo que se genera aquí es una instrucción para el *firewall* correspondiente a cada dirección IP que da el servidor DHCP. De esta manera, todos los paquetes que vengan de esa dirección se dirijirán hacia TocToc (Figura 4.2). En ese momento, el servidor Apache entra en acción y busca la página pedida en nuestra máquina. Si la página existe en nuestra máquina, este la sirve, pero si no es así, indicará un error 404. Como la idea del sistema es que salga nuestro portal siempre que se solicite una página, independientemente si existe o no, lo que se hace después es cambiar la página de error 404 para aquellas máquinas que provengan de la *wireless* a una página sencilla HTML que contenga una redirección a nuestro portal.

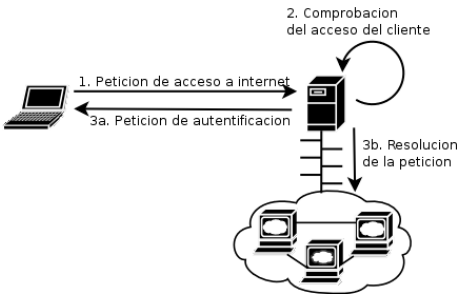

Figura 4.2: Captura de tramas por parte del sistema.

Por tanto, un cliente que esté situado en nuestra red local y pretenda acceder a Internet sin estar dado de alta en el mismo, debe de visualizar una pantalla como la mostrada en al Figura 4.3.

A esta página se le denomina *Splash Page*. Por *Splash Page* se entiende la página que aparece en el navegador en vez de la solicitada por el usuario, cuando este intenta acceder a Internet y el *Captive Portal* entra en funcionamiento. Generalmente, esta página normalmente se encarga de regular el acceso a usuarios registrados en el mismo o de mostrar un logotipo del proveedor.

A partir de esta pantalla, el usuario puede darse de alta, o en su caso, acceder con un usuario existente para después salir a Internet.

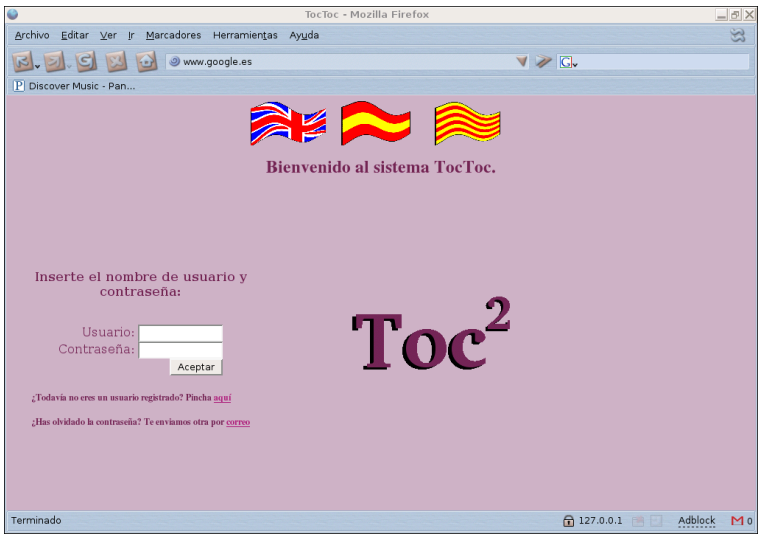

Figura 4.3: Pantalla de acceso a TocToc.

4.3. Registro de clientes.

El sistema debe guardar los clientes dados de alta en una base de datos. Esto incluyen algunos datos personales, el nombre de usuario y una contraseña de acceso. Además, junto a ellos se guarda cierta información relevante transparente a los usuarios que genera el sistema y le permite mantener el control sobre los mismos. Esto se hace mediante la base de datos MySQL. En la Tabla B.1 se muestran los campos del cliente que se guardan.

Estos datos permiten al sistema controlar lo siguiente:

- El *nick* y el *password* representa la forma de acceso al sistema para un usuario.
- El nombre, los apellidos y el país son información personal del usuario.
- El campo idioma guarda el idioma en el que se visualizará el portal. De esta forma un usuario evita tener que seleccionarlo cada vez entra en el sistema.
- Los campos *tmp_password* y *tiempo_tmp_password* son los campos que permiten a un usuario solicitar una contraseña cuando a este se le olvida su contraseña anterior. Esta contraseña se le enviará por correo y tiene una validez de 24 horas.
- Dispositivo, resolución y color permiten al sistema adaptarse mejor a las necesidades del cliente y mostrarse correctamente en pantallas pequeñas como las de una PDA.
- *Cityticket*, *vel_conexion* y *vel_subida* controlan la posibilidad de conexión de un cliente a Internet y a que velocidad realizará esta conexión.

Campo	Tipo	Significado
nick	VARCHAR(45)	El nombre de usuario.
pwd	VARCHAR(45)	La contraseña de acceso (encriptada).
cityticket	INTEGER	El nivel de acceso a Internet.
nombre	VARCHAR(45)	El nombre del cliente.
apellidos	VARCHAR(45)	Los apellidos del cliente.
pais	VARCHAR(45)	El país de procedencia del cliente.
email	VARCHAR(45)	Una dirección de contacto del cliente.
idioma	VARCHAR(10)	El idioma preferido de la página.
tmp_pwd	VARCHAR(45)	Contraseña temporal (encriptada).
tiempo_tmp_pwd	TIMESTAMP	Fecha de la contraseña temporal.
dispositivo	VARCHAR(15)	Tipo de dispositivo de acceso.
color	INTEGER	Color del dispositivo.
resolucion	INTEGER	Resolución del dispositivo.
vel_conexion	INTEGER	Velocidad de descarga del cliente.
vel_subida	INTEGER	Velocidad de subida del cliente.
IP	VARCHAR(15)	IP que está usando el cliente .
hora_conexion	TIMESTAMP	Hora a la que ha iniciado la conexión.
admin	INT(1)	Los derechos de administración.
fecha_fin_conexion	TIMESTAMP	Hora de la ultima desconexión.

Cuadro 4.1: Representación de los clientes en la base de datos.

- IP, *hora_conexion* y *fecha_fin_conexion* permiten al sistema saber si un cliente sigue conectado e imposibilitar así la conexión doble de un mismo cliente con dos equipos.
- El campo *admin* indica si ese usuario además tiene permisos de administrador.

4.3.1. Encriptación de contraseñas.

Como se puede observar la contraseña de usuario está encriptada. Para ello se ha utilizado el algoritmo de hashing MD5. Para añadir más seguridad y evitar un ataque por diccionario[1]

antes de encriptar se concatena a la contraseña una palabra cualquiera. Esta palabra puede ser modificada en cada sistema y viene determinada por el administrador del sistema (cuidado cuando se cambie esta palabra, pues el usuario administrador por defecto ya no podrá darse de alta en el sistema). Por último, para evitar que dos contraseñas iguales tengan la misma representación, se concatena el nombre del usuario. De esta forma no pueden existir dos contraseñas iguales en el sistema.

Como resultado de todo esto, aunque un administrador tenga acceso a la base de datos y la pueda visualizar, le será imposible deducir la contraseña de un usuario.

[1] Un ataque por diccionario consiste en generar un programa que contiene un listado de las contraseñas más utilizadas e ir contrastándolas una a una con las almacenadas en la base de datos.

4.4. Acceso posterior a Internet.

El funcionamiento del acceso a Internet, obliga a los clientes a darse de alta en el sistema. Una vez dados de alta, estos pueden contratar el acceso a Internet. Cuando ocurre esto, lo que tiene que ejecutar el cliente para permitir el acceso, es una orden que le indique al servidor que elimine la regla que impide su IP salir a Internet. Una vez eliminada la restricción, como el estado natural del sistema es permitir el redireccionamiento de paquetes, el cliente tendrá acceso a Internet.

Algorithm 2 Comandos que ejecuta un cliente para acceder a Internet.

```
#Variables y constantes.
define('IPT','/sbin/iptables');
$ip=getenv("REMOTE_ADDR"); #Se obtiene la IP del cliente.

#Se borran las ordenes que capturan los paquetes de la IP del cliente.
#El usuario apache tiene que tener permisos "sudo" sobre las IPtables.
exec('sudo '.IPT.' -t mangle -D POSTROUTING -d '.$ip);
```

Como se observa en el algoritmo 2 escrito en PHP, simplemente al activar la opción de conexión a Internet, se ejecuta un comando en el servidor que libera la regla del *firewall* que impide el acceso hacia el exterior. Al cliente se le muestra una ventana emergente que le indica que ya está conectado a Internet y que puede navegar normalmente.

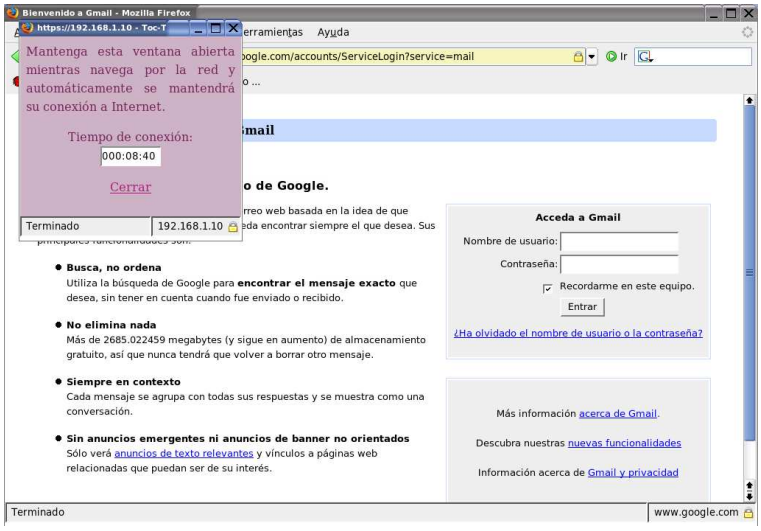

Figura 4.4: Ventana flotante de acceso a Internet.

Por último, conviene registrar el acceso del cliente en el *log* o en la base de datos del sistema y controlar que el mismo cliente pueda acceder varias veces a Internet en el mismo instante.

4.5. Velocidad de conexión de los clientes.

Hasta ahora tenemos control sobre que clientes pueden o no pueden conectarse a Internet. Sin embargo, cuando un cliente tiene acceso a la red, este lo hace con la única limitación de la propia velocidad de su tarjeta. En un sistema que se pretende cobrar o limitar el uso de un ancho de banda, es una situación no deseable. En la arquitectura TocToc esto se resuelve mediante la implantación del *Cityticket*.

4.5.1. Concepto de *Cityticket*.

Cityticket es el término que se ha designado a las distintas posibilidades de conexión que tiene un cliente. El nombre hace referencia a la compra de un ticket para contratar la conexión dentro de una ciudad. Representa las distintas tarifas que puede contratar un cliente, teniendo cada una de ellas unas velocidades de subida o bajada distintas.

La configuración por defecto del *Cityticket* viene definida en el archivo *configuracion.inc* dentro de los archivos PHP que se sirven al cliente y corresponde a la matriz *$array_velocidad_conexion*. La primera dimensión de esta matriz, representa los distintos *Citytickets* que existen en el sistema, que por defecto se han dejado en diez. La segunda dimensión, corresponde a las velocidades de subida y de bajada para cada uno de los *Citytickets* correspondientes.

4.5.2. Ejemplo de uso del *Cityticket*.

Una vez se implementa la capacidad de regulación de velocidades de los clientes, tendría que poder permitir el acceso a varios clientes con velocidades distintas. En la Figura 4.5, se puede observar una gráfica que representa la entrada de distintos clientes en el sistema en distintos intervalos de tiempo. Como se muestra, los clientes entran con una velocidad de 16 KB/s, 32 KB/s 128 KB/s y 64 KB/s respectivamente.

Como se muestra, cada uno de estos clientes cumple su velocidad de conexión de acuerdo con su *Cityticket*. Para implementar el *Cityticket* se ha utilizado la utilidad TC [6].

4.5.3. ¿Qué es TC?

TC es una herramienta que viene ya con todas las distribuciones Linux. Su nombre es el acrónimo de *traffic control* y permite generar un sistema de colas a las que se asigna unas propiedades, como por ejemplo, la velocidad de subida o bajada.

Como se puede ver en la figura 4.6, se pueden ir generando una infinidad de colas, que a su vez pueden colgar de otras colas consiguiendo una distribución del ancho de banda adecuado. Siempre existe una cola inicial, llamada *Root* que cuelga del dispositivo fijo. A partir de ahí se crea una segunda cola, que cuelga de la primera. Esta cola tiene como finalidad impedir que se consuma más de un porcentaje total del ancho de banda con un protocolo determinado. Por ejemplo, podemos indicar que solamente queremos que el protocolo IP consuma un 75 % del total del ancho de banda y así asegurar que otros paquetes importantes, como los ACK o las tramas ARP, tengan acceso a la red, permitiendo que los paquetes

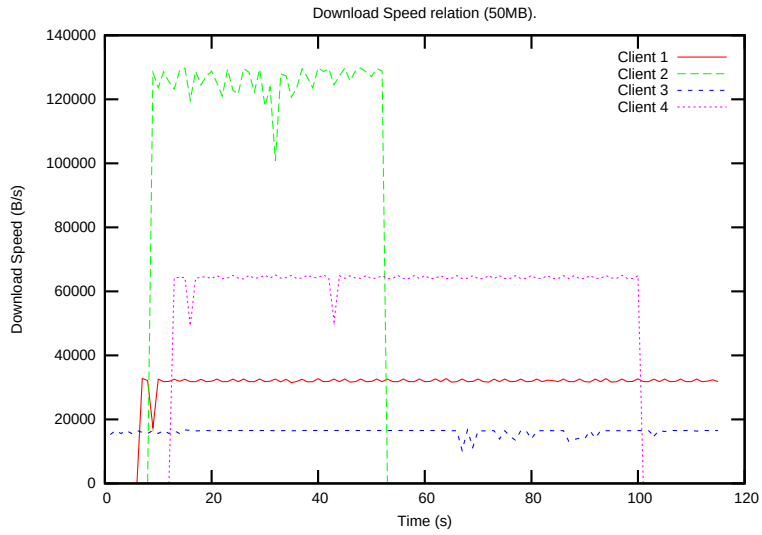

Figura 4.5: Ejemplo de distintos clientes en TocToc.

de confirmación lleguen a su destino y no se vuelvan a enviar innecesariamente, o la presencia de nuevos equipos en la red.

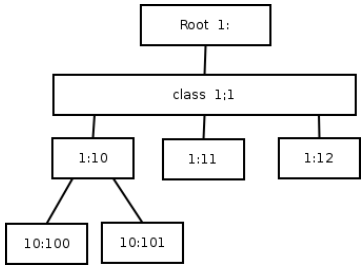

Figura 4.6: Jerarquía del funcionamiento de TC.

El resto de colas, se generan dependiendo del diseño del sistema. A cada cola se le puede asignar un consumo máximo y una prioridad determinada respecto a las demás colas. De esta forma se puede decidir que algunos programas, como por ejemplo los P2P, no saturen la red.

4.5.4. ¿Cómo se utiliza TC?

La mejor forma de responder a la pregunta, sería ir directamente a las páginas del *man* que dan una explicación extensa a esto o recurrir a la bibliografía que se presenta al final de este documento [6]. Sin embargo, aunque no se pretende tratar el tema con profundidad, a continuación se presenta un pequeño ejemplo explicativo de control mediante TC para dar a entender su funcionamiento:

En este ejemplo se va a separar el tráfico *ssh* del resto del tráfico. Para ello se generará una jerarquía de colas que permita dividir los paquetes de acuerdo

con la aplicación que los utilice.

tc qdisc add dev eth0 root handle 1: htb default 11

Esta línea genera la cola raíz y la asigna al dispositivo *eth0*. La cola por defecto, será la cola número 11, que la crearemos unas líneas más abajo. A esta cola irán todos los paquetes que no estén marcados.

tc class add dev eth0 parent 1: classid 1:1 htb rate 128kbit ceil 64kbit

Ahora, lo que se ha hecho es generar la cola 1:1 por debajo de la 1: y que sigue ligada al dispositivo *eth0* de manera que solamente permite una bajada de 128 Kb/s y una subida de 64 Kb/s. Esta cola será la principal de la cual colgarán el resto de las colas.

tc class add dev eth0 parent 1:1 classid 1:10 htb rate 100kbit ceil 50kbit prio 0

tc class add dev eth0 parent 1:1 classid 1:11 htb rate 80kbit ceil 40kbit prio 1

En este momento, existen dos nuevas colas. La primera cola (la 1:10) tiene mayor prioridad que la segunda (la 1:11) de forma que la segunda solamente alcanzará el máximo de 80Kb/s cuando la primera no necesite todo su ancho de banda asignado.

tc qdisc add dev eth0 parent 1:10 handle 100: sfq perturb 10

tc qdisc add dev eth0 parent 1:11 handle 110: sfq perturb 10

Se acaba de asignar el tipo de cola SFQ (*Stochastic Fairness Queueing*) que es una cola no demasiado precisa, pero que permite unos cálculos rápidos a la máquina. Existen distintos tipos de colas aceptadas por TC y se diferencian unas de otras en la manera de tratar los datos. Por ejemplo, existe la cola *pfifo_fast* de tipo FIFO y la cola TBF (*Token Bucket Filter*) mucho más precisa que la utilizada, pero con un coste computacional superior. Para mayor información, consultar la bibliografía [6].

tc filter add dev eth0 parent 1:0 protocol ip prio 1 handle 1 fw classid 1:10

tc filter add dev eth0 parent 1:0 protocol ip prio 2 handle 1 fw classid 1:11

Ahora se ha indicado que todos los paquetes pertenecientes al protocolo IP, que pasen por el dispositivo *eth0* y que tengan las marcas 1 y 2 en su cabecera, vayan a las colas 1:10 y 1:11 respectivamente. El resto de paquetes que no sean IP, no pasarán por ninguna cola, por tanto tendrán acceso libre a Internet.

Por último, se marcan los paquetes con los números 1 y 2 según se considere necesario para que vayan a las colas adecuadas. Esto se consigue mediante la ayuda de *IPtables*. Por ejemplo, si queremos que el *ssh* tenga mayor prioridad que cualquier otra cosa, tendríamos que asignarlo a la primera cola. Esto se realiza, controlando las peticiones que se dirijan al puerto *ssh* que es el número 22:

iptables -t mangle -A PREROUTING -p tcp -m tcp –sport 22 -j MARK –set-mark 0x1

iptables -t mangle -A PREROUTING -p icmp -j RETURN

Acabamos de marcar todos los paquetes *ssh* para que vayan a la cola con mayor prioridad. Mientras que el resto de paquetes de tipo IP, irán a la cola por defecto que es la 11.

4.5.5. Aplicación de TC en el sistema TocToc.

Una vez ya se ha mostrado una idea básica del funcionamiento del TC, se presenta la parte del código que se ha elaborado en el sistema para hacer uso de esta herramienta. Está dividido en dos partes. La parte del servidor (algoritmo 3) prepara las colas iniciales, tanto de subida como de bajada, en los dispositivos adecuados. La parte que activa el cliente cuando ejecuta la orden de conectarse a Internet (algoritmo 4) genera la cola con la velocidad permitida para ese cliente y asigna todos sus paquetes a la misma.

El algoritmo 3, como se muestra, prepara al sistema para la futura conexión de clientes. Para ello, lo primero que hace es limpiar las reglas anteriores TC como medida preventiva. Después, se generan las colas raíz que se aplican a cada dispositivo elegido por el sistema. Como se puede observar se generan dos colas raíz, una de subida y otra de bajada. De esta forma se pretende conseguir un control de ambas velocidades. En este algoritmo solamente se generan las colas del dispositivo, ya que todavía no sabemos cuantos clientes accederán al sistema.

En el segundo algoritmo (el número 4) se generan las colas de acuerdo con los privilegios del cliente. Así, una vez un cliente accede a Internet, se generan las colas de acuerdo con los límites de velocidad que tiene. Esto se realiza cada vez que un cliente pulsa la opción de conectar a Internet y se destruyen cada vez que un cliente cierra la venta de navegación que mantiene la conexión a Internet.

4.6. Liberación de clientes una vez finalizado la tarea.

Pueden existir dos formas de dar de baja un cliente. Que el cliente voluntariamente se desconecte pulsando la opción correspondiente en la ventana emergente, o que el cliente pierda su conexión con la red y deje de tener acceso físico al servidor.

4.6.1. Desconexión voluntaria del cliente.

Si un cliente se desconecta voluntariamente del sistema, lo único que tiene que hacer el sistema es indicar al *firewall* que la IP utilizada deja de tener acceso a Internet. Después de ello se desactivará el uso del cliente en la base de datos para que el cliente pueda volver a entrar a Internet.

La ventana de cierre del cliente, ordena la ejecución de una instrucción que hace que el sistema vuelva a capturar todas las peticiones del cliente que no vayan a un servidor permitido.

Algorithm 3 Control de velocidad mediante TC por parte del sistema.

```
#Este algoritmo está incluido dentro del script TocToc.sh.

#Variables.
TC=/sbin/tc
ETH="eth1"
WIR="wlan0"
RATEUP=3750
RATEDOWN=7500

#Borramos cualquier cosa que haya de antes.
$TC qdisc del dev $ETH root 2>/dev/null >/dev/null
$TC qdisc del dev $ETH ingress 2>/dev/null >/dev/null
$TC qdisc del dev $ETH root 2>/dev/null >/dev/null
$TC qdisc del dev $WIR root 2>/dev/null >/dev/null
$TC qdisc del dev $WIR ingress 2>/dev/null >/dev/null
$TC qdisc del dev $WIR root 2>/dev/null >/dev/null

#Por que conexión pasa el tráfico por defecto. La cola 2 es la cola de los paquetes
que salen a Internet no clasificados (ACKs).
$TC qdisc add dev $ETH root handle 10: htb default 2
$TC qdisc add dev $WIR root handle 11: htb default 3

#Generación de la jerarquía. La 10:1 es la que agrupa a todos y la 10:2 la cola
por defecto. Ídem para 11:1 y 11:20
$TC class add dev $ETH parent 10: classid 10:1 htb rate ${RATEUP}kbit ceil
${RATEUP}kbit
$TC class add dev $WIR parent 11: classid 11:1 htb rate ${RATEDOWN}kbit
ceil ${RATEDOWN}kbit

$TC class add dev $ETH parent 10:1 classid 10:2 htb rate 16kbit ceil $[80*$RA-
TEDOWN/100]kbit
$TC qdisc add dev $ETH parent 10:2 handle 2: sfq perturb 10
$TC filter add dev $ETH parent 10: protocol ip handle 2 fw classid 10:2

$TC class add dev $WIR parent 11:1 classid 11:3 htb rate 8kbit ceil $[80*$RA-
TEUP/100]kbit
$TC qdisc add dev $WIR parent 11:3 handle 22: sfq perturb 10
$TC filter add dev $WIR parent 11: protocol ip handle 4 fw classid 11:3
```

4.6.2. Desconexión involuntaria.

Un cliente puede perder su conexión de red de dos formas: una forma física debido a la pérdida de señal de la red *wireless*, y otra debido a que el cliente cierra incorrectamente la ventan flotante que mantiene la conexión con el servidor. En ambos casos, el servidor simplemente deja de tener contacto con el cliente y lo considera desconectado después de cierto tiempo. La forma de desconexión es similar a la versión voluntaria. La diferencia es que, en vez de ser el cliente el que pide la desconexión, es el servidor el que debe percatarse de ello. Sin embargo,

Algorithm 4 Control de velocidad mediante TC por parte del cliente.

// Este algoritmo se utiliza en la función *da_internet_usuario($conexion)* incluido en el archivo *redWireless.inc*.

```
//Variables y constantes.
//$ip y $ip_array contiene la IP del cliente que se conecta. La última dividida en un vector.
//$velocidad_conexion_Kb y $velocidad_subida_Kb son los valores obtenidos de la base de datos correspondientes al cliente que ejecuta esto.
//Redes de entrada y salida.
define('RED_WIRELESS','wlan0');
define('RED_INTERNET','eth1');
//Localización de los binarios.
define('TC','/sbin/tc');
define('IPT','/sbin/iptables');


//Programamos la velocidad de descarga del usuario.
exec('sudo '.TC.' class replace dev '.RED_WIRELESS.' parent 11:1 classid 11:'.$ip_array[3].' htb rate 1kbit ceil '.$velocidad_conexion_Kb.'kbit prio 2');
exec('sudo '.TC.' qdisc replace dev '.RED_WIRELESS.' parent 11:'.$ip_array[3].' handle '.$ip_array[3].': sfq perturb 10');
exec('sudo '.TC.' filter replace dev '.RED_WIRELESS.' protocol ip parent 11: handle '.$ip_array[3].' fw classid 11:'.$ip_array[3]);
exec('sudo '.IPT.' -t mangle -D POSTROUTING -d '.$ip);
exec('sudo '.IPT.' -t mangle -A POSTROUTING -d '.$ip.' -j MARK –set-mark '.$ip_array[3]);

//Programamos la velocidad de subida.
exec('sudo '.TC.' class replace dev '.RED_INTERNET.' parent 10:1 classid 10:'.$ip_array[3].' htb rate 1kbit ceil '.$velocidad_subida_Kb.'kbit prio 2');
exec('sudo '.TC.' qdisc replace dev '.RED_INTERNET.' parent 10:'.$ip_array[3].' handle '.$ip_array[3].': sfq perturb 10');
exec('sudo '.TC.' filter replace dev '.RED_INTERNET.' protocol ip parent 10: handle '.$ip_array[3].' fw classid 10:'.$ip_array[3]);
exec('sudo '.IPT.' -t mangle -D PREROUTING -s '.$ip);
exec('sudo '.IPT.' -t mangle -A PREROUTING -s '.$ip.' -j MARK –set-mark '.$ip_array[3]);
```

Algorithm 5 Desactivación de un cliente de forma voluntaria.

```
//Constantes
define('IPT','/sbin/iptables');
define('IP_GATEWAY','192.168.1.10');

//El usuario apache tiene que tener permisos "sudo" sobre las IPtables.
exec('sudo '.IPT.' -t nat -A PREROUTING -p TCP -s '.$ip.' -j DNAT –to-destination '.IP_GATEWAY,$retval);
```

debido al funcionamiento de una página web en PHP, el servidor difícilmente puede saber si el cliente está conectado o no en un intervalo de tiempo pequeño. Para facilitar la tarea, un cliente cada cierto tiempo, actualiza su estado en un fichero que tiene como nombre su IP y que reside en la máquina servidor. Este fichero tiene cuatro estados:

- El cliente acaba de actualizar su estado.
- El cliente no ha accedido recientemente, se le otorga una segunda oportunidad.
- No se tienen noticias del cliente recientemente, por lo que se desconecta.
- El cliente no está conectado, por lo que no se hace nada.

Este funcionamiento se muestra en la Figura 4.7.

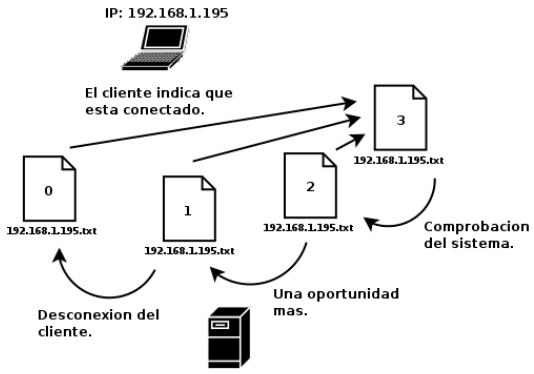

Figura 4.7: Funcionamiento del control de clientes.

Para ello, el algoritmo 6 se queda activado en *background* buscando aquellos clientes que no actualizan su estado.

Este algoritmo busca solamente a aquellos posibles clientes dentro del rango de IPs que controla TocToc en vez de todo el rango posible. De esta forma se regula el número de clientes que entran al sistema mediante esta arquitectura y se carga menos el sistema.

4.7. Servidores permitidos a todos los clientes.

Por diversos motivos, puede existir un conjunto de servidores a los que sí que esté permitido acceder aunque el cliente no esté dado de alta. Esto es una realidad que se observa en multitud de ocasiones:

- Un *Captive Portal* de un aeropuerto ofrece gratuitamente información de todos sus vuelos, embarques y demás servicios de la instalación. Esto rara vez esta alojado en una misma computadora.
- Si el *Captive Portal* se aplica al turismo, se puede dar acceso a las páginas oficiales del lugar, como la web del ayuntamiento, museos, etc.

Algorithm 6 Búsqueda de clientes desconectados.

```
#Constantes.
PWLAN="192.168.1"
TOCTOC="10"
DHCPmin="195" #Para el cliente 192.168.1.195
DHCPmax="205" #Hasta el cliente 192.168.1.205
TIEMPO="20"
DIR_CLIENTES="/TocToc_clientes"
PUERTOS_ACEPTADOS="9400:9800"

while (true) do
for ((i = $DHCPmin; i <= $DHCPmax; i++)) do
numero=$(head -1 $DIR_CLIENTES/$PWLAN.$i.txt);
fecha=$(date + %d" " %m" " %Y" " %T);
case $numero in
"3") #El cliente acaba de estar, bajamos en uno el grado.
echo 2 >$DIR_CLIENTES/$PWLAN.$i.txt;
;;
"2") #El cliente no ha respondido, segunda oportunidad.
echo 1 >$DIR_CLIENTES/$PWLAN.$i.txt;
;;
"1") #El cliente ya no esta, se desconecta de Internet.
iptables -t nat -A PREROUTING -p TCP –dport ! $PUERTOS_ACEPTADOS
-s $PWLAN.$i -j DNAT –to-destination $PWLAN.$TOCTOC;
echo 0 >$DIR_CLIENTES/$PWLAN.$i.txt;
;;
*) #No se hace nada para un cliente desconectado.
;;
esac
done
sleep $TIEMPO;
done
```

- Si se da un servicio de VoIP, se debe de poder descargar algunos de los programas que lo permiten. Estos programas pueden estar alojados en un servidor FTP externo.

- Etcétera.

Existen dos formas de programar estos servidores externos permitidos: mediante puertos o mediante IPs. Cada una de las opciones tienen ventajas y desventajas, por ello se van a tratar por separado.

4.7.1. Acceso mediante puertos.

El acceso mediante puertos consiste en disfrazar la petición de una dirección web con un puerto que más tarde el sistema TocToc interpretará como la petición de una dirección web determinada. Para ello, se completa el algoritmo 1 presentado anteriormente modificando la línea:

$IPT -t nat -A PREROUTING -p tcp -s $PWLAN.$i -j DNAT –to-destination $PWLAN.$TOCTOC

Por la siguiente:

$IPT -t nat -A PREROUTING -p tcp –dport ! $PUERTOS_ACEPTADOS -s $PWLAN.$i -j DNAT –to-destination $PWLAN.$TOCTOC

Siendo *$PUERTOS_ACEPTADOS* una variable que contiene el intervalo de puertos que se deseen reservar. Por defecto, esta variable vale "9400:9800". Como se puede observar, esta modificación indica que la regla se aplica para todos los paquetes TCP que no se dirijan a los puertos indicados.

Hay que señalar, que realmente no importa la dirección web que se indique en el portal (ver sección 5.5.6), sino la asociación que tiene el sistema con dicho puerto. Es decir, realmente da igual llamar a UPV:9400 `http://www.upv.es:9400` si después el puerto 9400 esta asociado con Google. El cliente seguirá recibiendo la página de Google.

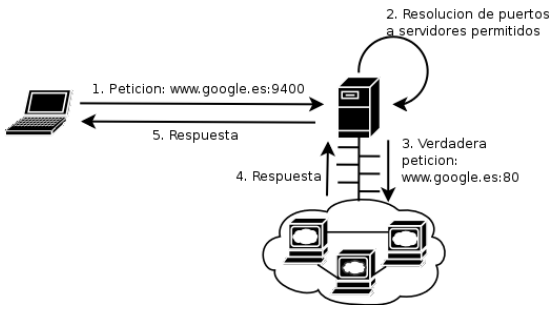

Figura 4.8: Funcionamiento de acceso por puertos.

4.7.2. Acceso mediante IPs.

Otra forma de dar de alta una petición es dar de alta en el sistema su IP. Esto se consigue de la forma indicada en el algoritmo 7. Estas reglas permiten el acceso a un listado de IPs determinado, y si están colocadas antes que las reglas de prohibición, permiten el acceso a los servidores.

Algorithm 7 Código para dar paso a una *IP* a través del sistema.

```
#Variables.
IPT=/sbin/iptables
IPS_ACEPTADAS="64.233.183.147 64.233.183.99 ..."

for i in $IPS_ACEPTADAS
do
$IPT -t nat -A PREROUTING -p tcp -d $i -j ACCEPT
done
```

El *firewall* de Linux busca la primera regla que cumpla las condiciones dadas, de manera que antes de prohibir el acceso al cliente, buscará si es una IP permitida y en su caso, realizará la petición externa. En la figura 4.9 se representa el funcionamiento del mismo.

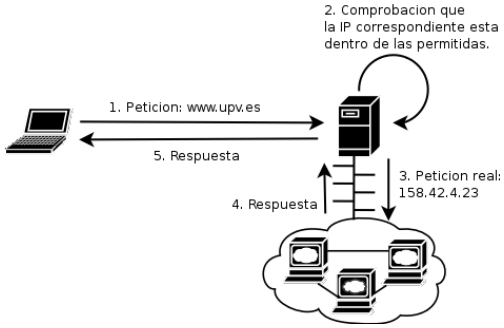

Figura 4.9: Funcionamiento de acceso mediante *IP*.

4.7.3. Diferencias entre acceso mediante IPs y mediante puertos.

La diferencia entre el acceso mediante IPs y el acceso mediante puertos, aparte de la implementación interna del mismo, consiste en que, una máquina dentro de la red inalámbrica podrá acceder a un servidor dado de alta mediante IPs, sin pasar por el portal TocToc. Sin embargo, para acceder a un servidor permitido por puertos, a no ser que se conozca con que puerto está relacionado, se deberá de pasar por el portal que es el que indicará con un enlace como acceder al mismo. Una desventaja de este último, es que si un servidor externo tiene enlaces absolutos en vez de relativos, se perderá la información del puerto de acceso y por tanto el cliente acabará de nuevo en la página inicial de TocToc.

4.7.4. Almacenamiento del listado de servidores gratuitos.

El sistema permite almacenar los servidores gratuitos de dos formas. La primera consiste en la propia configuración del script del servidor. Así se puede activar, si el acceso es mediante IP, una simple variable en el *script* que incluye todas las IPs a las que se quiere dar acceso. Esta variable se puede observar en el algoritmo 7 y se muestra de esta forma: *IPS_ACEPTADAS="64.233.183.147 64.233.183.99 ..."*. Esta variable contiene una lista de IPs separadas por espacios. Si el acceso es mediante puertos, habrá que programar cada regla de forma manual como se ha indicado anteriormente.

La otra forma de almacenar los servidores gratuitos, es mediante la base de datos MySQL. El sistema acepta que se inserten estos servidores mediante las herramientas de control de las que dispone. Esto se puede observar claramente en la sección 5.5.6. Internamente, el sistema memoriza en la base de datos una estructura como la que se muestra en la tabla B.5.

Esta estructura almacena ya tanto los servidores definidos por puertos, como los definidos exclusivamente por IPs. El sistema después reconocerá cada uno

nombre	puerto	direccion	ip	dest_port
Google	9400	http://www.google.es	72.14.207.104	80
UPV		http://www.upv.es	158.42.4.23	80
...	...	...	...	...

Cuadro 4.2: Almacenamiento de los servidores por MySQL.

de ellos dependiendo si el campo "puerto" está vacío o no.

No existen diferencias de funcionamiento entre las direcciones en la base de datos y las escritas en el propio script.

4.7.5. Planificación de los servidores gratuitos.

Hay que tener cuidado a la hora de decidir a que servidores se da acceso gratuito. Se recomienda evitar aquellos que tengan opciones de descarga (como RedIris) ya que los clientes acceden a estos sin control de ancho de banda, es decir, obtendrán toda la velocidad de descarga posible. Esto puede desembocar en un consumo excesivo de la red impidiendo el acceso a las clientes con cuentas de pago. Puede ser interesante colocar servidores derminados con algunos programas en concreto, como por ejemplo Skype, si se pretende dar servicios de VoIP; pero se recomienda que sean servidores propios para controlar en todo momento el número y tamaño de los archivos descargables, evitando el abuso de este servicio.

4.8. Control de dispositivos.

El sistema TocToc permite al diseñador del portal intentar adaptar la visualización del mismo según el dispositivo al que se accede al sistema. Puede ocurrir, que el cliente que ha contratado el servicio acceda con una resolución muy inferior a la recomendada o con una gama de colores no ideal. Como consecuencia de ello, el portal puede aparecer un poco desajustado y con una visualización no deseada. Para ello, cuando un usuario se da de alta, se le pide algunos datos relativos al dispositivo que usa al acceder. Esta información puede ser utilizada para intentar solventar algunos detalles de visualización. Por ejemplo, si un cliente usa una PDA en blanco y negro para conectarse en el sistema, puede resultar un desastre mostrarle una presentación Flash al inicio de la página. Se le podría, sin embargo, mostrar el logotipo del sistema de una forma simplificada.

Los datos del dispositivo se guardan en la base de datos y en la sesión del usuario, de forma que puedan ser consultados en cualquier momento. El cómo se hace esto se puede observar en las tablas B.2, B.3 y B.4.

dispositivo
Portatil
PDA
...

Cuadro 4.3: Control de dispositivos.

La primera de las tablas contiene un listado de todos los dispositivos que se piensan que se utilizarán para acceder al sistema. La utilidad de esto, será principalmente administrar y dividir las demás tablas de acuerdo con el dispositivo, por ejemplo, algunas resoluciones estarán solamente disponibles para ciertos dispositivos.

tipo	dispositivo	id
800 x 600	Portatil	1
1024 x 768	Portatil	2
320 x 320	PDA	3
...	...	4

Cuadro 4.4: Control de resoluciones.

La tabla B.3 guarda las distintas resoluciones posibles para cada dispositivo. En el ejemplo mostrado, un dispositivo portátil puede elegir una resolución de 800 x 600 o 1024 x 768, mientras que una PDA solamente puede elegir una resolución 320 x 320. El campo id es la clave principal de cada elemento, y será por tanto lo que se memorizará cuando cada cliente haga su selección.

profundidad	dispositivo	id
16bits	Portatil	1
2bits	PDA	2
...	...	3

Cuadro 4.5: Control de colores.

En la tabla B.4 se agrupan los colores para cada uno de los dispositivos. De esta forma se puede hacer, por ejemplo, que una PDA solamente pueda visualizar el portal de forma monocroma, mientras que un portátil puede tener una gama de colores más amplia.

4.9. Registro de acceso: Log del sistema.

Como en la mayoría de sistemas, es conveniente tener un registro de todas las acciones que se llevan a cabo. En el sistema TocToc, se guarda en el registro la información del usuario cada vez que este se conecta a Internet en el sistema o cada vez que este se desconecta, ya sea voluntariamente o involuntariamente.

Cada línea del *log*, guarda el nombre de usuario, si realizó una conexión o una desconexión en el sistema, la IP desde la que realizó dicha conexión o desconexión, la fecha de la misma y si, en el caso de que fuera una desconexión, fue hecha de forma forzosa, entendiendo como desconexión forzosa cuando el servidor dejara de tener noticias del cliente y lo da de baja en el sistema.

Este archivo, permite controlar el tiempo de conexión total de un usuario y ver todos los accesos durante ese periodo. Para más información, dirigirse a la sección 5.5.2.

Por agilización de control de *log*, este conserva solamente las entradas de un periodo determinado, como por ejemplo, un mes. Esto viene determinado

por la constante *TIEMPO_LOG* definida en el archivo *configuracion.inc* del sistema. Todas las entradas anteriores a este periodo, se traspasan a un *backup* por motivos de seguridad.

4.10. Multilingüismo.

4.10.1. Estructura básica.

El sistema TocToc incluye unas funciones en lenguaje PHP para poder añadir multilingüismo al portal. Una vez incluidas estas funciones en el sistema, se permite añadir varios nuevos idiomas de una forma sencilla. La forma diseñada para ello, obliga a generar dos tipos de archivos para cada página web: el HTML básico que contiene la estructura y el XML asociado que contiene todo el texto en los distintos idiomas deseados. De esta forma, añadir un nuevo idioma solamente implica añadir una nueva linea para cada párrafo que contenga la traducción a dicho idioma.

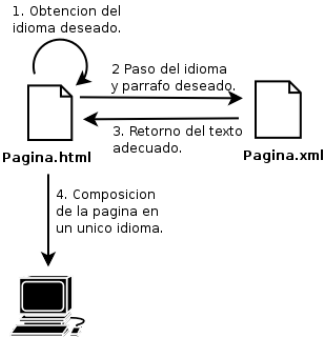

Figura 4.10: Funcionamiento del Multilingüismo.

4.10.2. Ejemplo de diseño.

Lo explicado anteriormente, se puede ver de una forma mas técnica en el ejemplo 14, en donde se muestra la estructura de un documento sencillo con dos idiomas: español e inglés. En este ejemplo se ve la estructura típica de un documento XML, con la cabecera correspondiente y las etiquetas del documento. Cada etiqueta adicional corresponde a un párrafo de texto, y esta a su vez se divide en los dos idiomas que se han utilizado.

Algorithm 8 Ejemplo de página XML con diversos idiomas.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<toctoc>
  <hola_mundo>
    <esp>Hola mundo</esp>
    <eng>Hello Word</eng>
  </hola_mundo>
</toctoc>
```

Esta página XML, cuando sea llamada por la página web correspondiente, de la forma indicada en el algoritmo 9 obtendrá el texto deseado según el idioma seleccionado.

Algorithm 9 Ejemplo de página HTML *ejemplo.html* con uso multilingüe.

```
<html>
<head>
<title>Pagina de ejemplo</title>
<?php
include("include/XML.inc");
$fichero="idiomas/ejemplo.xml";
abreXML($fichero,$arr_vals, $arr_index);
?>
</head>
<body>
<?php obtieneCampoXML($arr_vals, "HOLA_MUNDO", "ESP", $texto);
echo $texto; ?>
</body>
</html>
```

Como se puede ver en ese ejemplo, para el correcto funcionamiento hay que realizar los siguientes pasos:

1. Incluir las funciones necesarias (*include("include/XML.inc");*). En este archivo están todas las funciones PHP diseñadas con el propósito de seleccionar el idioma adecuado.

2. Incluir la página que contiene el texto en los diversos idiomas *($fichero="idiomas/ejemplo.xml";* y *abreXML($fichero,$arr_vals, $arr_index);*). Cada una de la páginas web debe tener una página XML asociada. Con estas dos instrucciones se le indica que página es esta y le ordena a PHP cargarla en memoria en forma de matriz (llamada en este caso *$arr_vals*).

Incluir para cada párrafo, la petición en concreto (*obtieneCampoXML($arr_vals, "HOLA_MUNDO", "ESP", $texto);*). Esta última función busca el párrafo etiquetado como "*hola_mundo*" dentro de la matriz *$arr_vals* usando el idioma español y lo guarda en la variable *$texto*. Solamente queda imprimir el contenido de esta variable en donde se quiera situar el texto.

Este último paso habrá que repetirlo en cada uno de los textos que aparecen en la página web. Desde los párrafos normales, hasta las etiquetas de los botones de los formularios.

4.10.3. Activación de los idiomas.

Como se ha dicho anteriormente, cada idioma se debe de introducir en cada una de las páginas XML en donde se escribe el texto a mostrar. Sin embargo, el hecho de que existan estas líneas no implica que el usuario pueda seleccionar ese idioma en concreto. Pueden ir incluyéndose poco a poco un nuevo idioma y solamente activarlo cuando este está listo. Para ello, se debe permitir que los usuarios puedan optar a seleccionar dicho idioma. Para que el sistema añada

todas las opciones necesarias, tanto al inicio de la página, como en el perfil del usuario, se debe de completar la tabla de MySQL que aparece representada en la tabla número B.6.

id	idioma	bandera
ENG	English	idiomas/banderas/ingles.gif
ESP	Español	idiomas/banderas/castellano.gif
VAL	Valencià	idiomas/banderas/valenciano.gif

Cuadro 4.6: Gestión de idiomas mediante MySQL.

Como se puede observar, la primera columna tiene que corresponder con el código que se empleó al diferenciar los idiomas en las etiquetas XML. La segunda columna será el nombre de dicho idioma (aconsejablemente, en el idioma del mismo) y la tercera columna, incluye el camino a la imagen que contiene la bandera del mismo.

4.10.4. Algunos detalles a tener en cuenta en este diseño.

Un inconveniente que tiene esta forma de representar el texto, reside en los carácteres propios del lenguaje español. Existe un conjunto de carácteres que en principio no están soportados por el sistema de codificación estándar. Estos lenguajes incluyen los acentos, la "ñ" y algunos carácteres especiales como "¿" y "¡". HTML soluciona esto escribiendo esos caracteres con unas etiquetas especiales que no incluyen esos caracteres. Por ejemplo, para la "á" se escribiría "*á*". Sin embargo, esta etiqueta, tiene un carácter especial que es incompatible con XML (es un carácter reservado). Este carácter es "&" por lo que este a su vez se tiene que volverse a codificar como "*&*". Una forma mas sencilla de ver esto es mediante la figura 4.11. En esta figura se puede observar los pasos que se siguen hasta formar un símbolo de interrogación de inicio de oración.

Figura 4.11: Codificación de los carácteres especiales latinos.

Este detalle de codificación es muy importante si en un futuro se realiza una aplicación que permita introducir los diferentes idiomas de una forma más visual que la programación mediante un editor de texto.

4.11. Parada del servicio.

Como toda aplicación, TocToc puede ser detenida si se considera necesario. Para ello, utilizando la opción de administración correspondiente en el portal o ejecutar en la línea de comandos *"sh TocToc.sh stop"* bastará para detener el

sistema. La máquina se convertirá en un *Gateway* convencional que permitirá el acceso a todos los clientes de forma normal. Si se pretende modificar el comportamiento de la máquina para que realice otra acción, como que prohiba el acceso a todas las máquinas, habrá que cambiar la sección "stop" que aparece en el *script* TocToc.sh y modificar las reglas del *firewall* pertinentes.

4.12. Facilidades en la integración con otros sistemas.

Esta arquitectura esta realizado en su mayoría en PHP. Las características de acceso están realizadas mediante funciones almacenadas en distintos archivos que se pueden incluir en otros proyectos. Esto significa que para obtener toda la funcionalidad del sistema dentro de una página web, solamente se tiene que incluir la página que contiene las funciones (mediante la función *include* de PHP) y utilizar las funciones correspondientes en donde se considere necesario.

La hoja de estilos y las distintas variables del fichero *configuracion.inc* permiten armonizar el sistema TocToc con cualquier página web que se quiera realizar. Mientras, las base de datos MySQL pueden permanecer aparte de cualquier otra base de datos que se quiera añadir después, o ser ampliadas de forma que den una funcionalidad mayor. Un ejemplo de esto se puede ver en el capítulo 8.

Esto no dice que la fusión sea sencilla. De hecho, está reservada a personas con conocimientos de PHP que entienden como funciona el sistema aunque sea de una forma superficial. Precisamente por esto, se aconseja trabajar a partir del sistema TocToc en vez de fusionarlo con otra páginas. En el diseño se ha dado a TocToc tiene una estructura un tanto esquemática, de forma que no resulte demasiado complicado amoldarlo a los gustos de cada programador.

Capítulo 5

Arquitectura del portal. Ejemplos de uso.

En esta sección se presenta una muestra del funcionamiento del portal. Cada explicación viene acompañada de capturas de pantalla que muestran los pasos a seguir en el sistema. Esta sección pretende mostrar y a su vez enseñar el manejo de TocToc.

5.1. Presentación del portal.

Supongamos que el sistema esta activo y que un usuario conectado a la red pretende acceder a Internet. Este usuario, escribe una *URL* en su navegador y espera obtener el resultado esperado. Sin embargo, el *Captive Portal* realizará su trabajo y lo impedirá. Lo primero que aparece al intentar acceder a Internet, es la página inicial del sistema TocToc. Esta página, puede ser personalizable de acuerdo con las necesidades del sistema donde se agregue. Lo único fundamental de esta página, es el formulario de entrada y las banderas de idiomas.

En este ejemplo, arriba del todo, aparece una bandera por cada idioma disponible en el portal. La selección de una de las banderas hará cambiar todo el texto del mismo al idioma correspondiente.

A la derecha, aparece el espacio reservado para el logo del sistema, una presentación o cualquier otro recurso multimedia que se crea necesario. Una buena idea sería avisar aquí al usuario de lo que le está ocurriendo y de por qué no se visualiza la página solicitada.

A la izquierda aparecen la parte de autentificación de usuarios. Un enlace inferior, permite registrarse si todavía no se ha hecho, y otro enlace por debajo de este último, permite recuperar la contraseña del usuario enviándola al *e-mail* que este indico al darse de alta.

Hay que tener en cuenta un detalle muy importante para el funcionamiento del sistema, y es que, como en todos los *Captive Portal* el cliente no debe escribir en la *URL* de su navegador directamente la dirección del propio *Captive Portal*, ya que cuando este acceda a Internet a través de él, le enviará a la página web solicitada, que será el propio *Captive Portal*. Como al entrar en TocToc, lo primero que se hace es borrar las sesiones de PHP anteriores por seguridad, se pierden todos los datos de usuario y este tiene que volver a darse de alta.

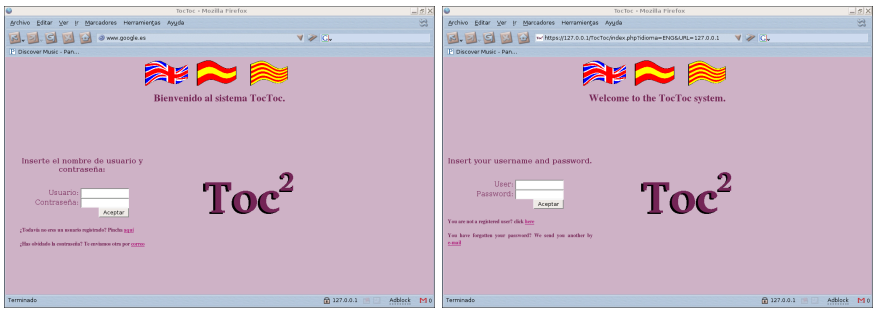

Figura 5.1: Presentación del sistema TocToc en diversos idiomas.

A su vez, si vuelve a acceder a Internet, vuelve a ocurrir lo mismo. Esto puede generar mucha confusión en un usuario y este puede llegar a pensar que es un fallo del sistema.

5.2. Registro de usuario.

Si accedemos a partir de la página inicial a la creación de un nuevo usuario, nos aparecerá una página como la Figura 5.2 que contiene distintas partes. La primera de todas, no es más que el título de la página, seguida de un pequeño campo donde el sistema muestra los errores que pueden ocurrir. En el ejemplo mostrado, aparece un error que indica que el usuario introducido ya existe. Como se puede observar, el color del error es distinto al color del texto normal.

A continuación se presenta el formulario de inscripción. Con colores más llamativos (dependiendo de la configuración que se le haya dado al sistema) aparecen los campos obligatorios. Un campo obligatorio es aquel que debe ser rellenado para poder darse de alta. Con color de texto normal, aparecen los campos opcionales, que son aquellos que pueden dejarse en blanco si se desea. El último campo, corresponde al idioma preferido para la página. Este campo permite que la página se muestre siempre en el idioma que el usuario seleccionó en el momento se registró en el sistema.

Una vez escritos los datos del usuario, el sistema solicita unos datos relativos al dispositivo con el que se accede al mismo. En la Figura 5.3 se muestra como un usuario ha seleccionado que accede al portal mediante un ordenador portátil que tiene una profundidad de color de 16 bits y una resolución de 1024 x 768. Estos datos se recogen, ya que dependiendo del dispositivo con el que se acceda y de su resolución, el sistema puede adaptar su formato para que se muestre de la mejor forma posible.

5.3. Una vez dentro del sistema.

Un cliente solamente puede acceder al sistema entrando con un usuario registrado. Cada usuario puede tener distintos privilegios que le pueden se asignados por el administrador del sistema. Así los usuarios se dividen en usuarios convencionales, usuarios con acceso a Internet y administradores.

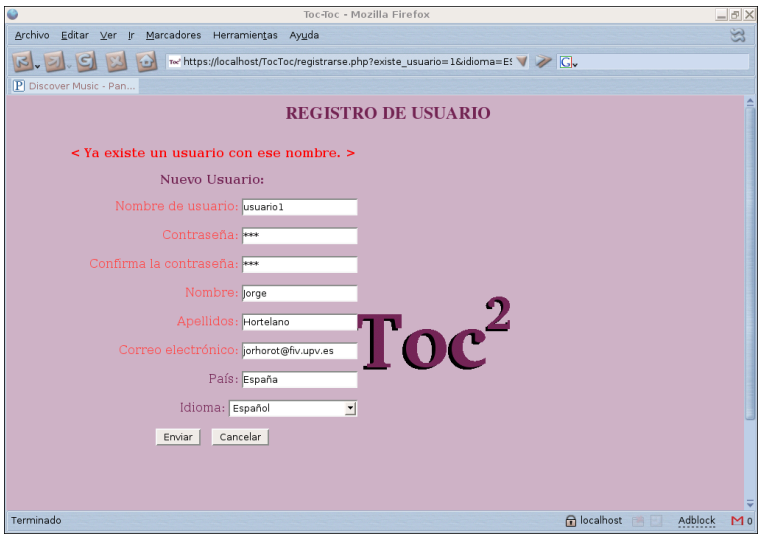

Figura 5.2: Página de registro del sistema.

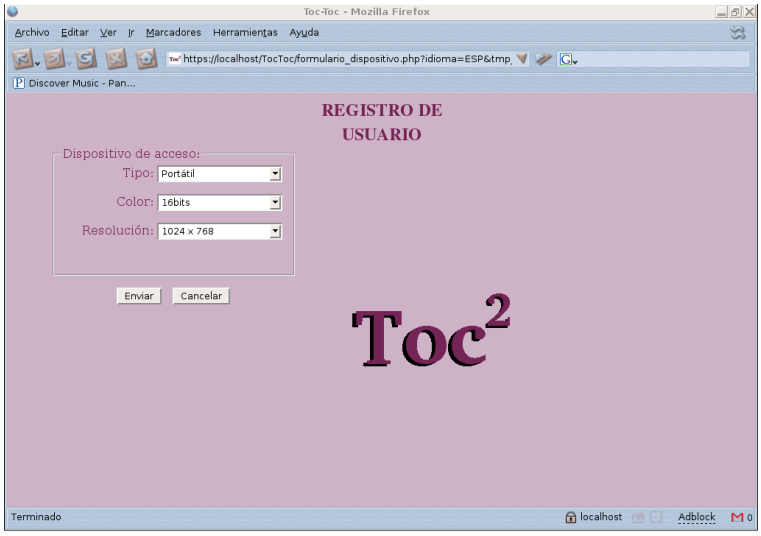

Figura 5.3: Configuración del dispositivo de acceso.

5.3.1. Lo más básico a lo que accede un usuario.

Una vez dado de alta el usuario, este puede acceder al sistema introduciendo su nombre de usuario y su contraseña. Lo que verá el usuario, una vez acceda, depende en parte de los servicios que se quieran ofertar gratuitamente y de aquello a lo que haya pagado el cliente para acceder. En la Figura 5.4 se muestra lo primero que se encontrará un cliente de forma gratuita. En el menú superior se muestran algunos datos de interés, como el nombre de usuario con el que se ha accedido al sistema, la fecha de acceso al mismo y el nivel de acceso, que en

este caso es el más básico disponible. Al lado derecho aparece una opción para salir del sistema y cerrar la cuenta de usuario.

Un menú a la izquierda, muestra las opciones a las que puede acceder el usuario. En este caso, solamente puede acceder a la modificación de su perfil, por si desea cambiar sus datos personales; puede acceder también a los servidores a los que el sistema le permita el acceso de forma gratuita (cuando se selecciona, aparece una lista con todos ellos) y por último, puede acceder a la ayuda del sistema, en donde el usuario puede consultar la información del sistema que le explicará, por ejemplo, como solicitar su acceso a Internet.

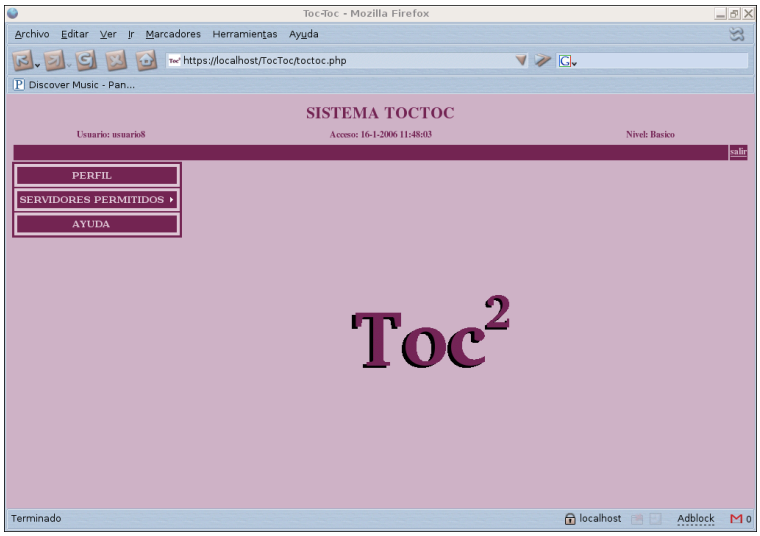

Figura 5.4: El sistema visto por un cliente.

Un ejemplo más completo se muestra en el capítulo 8, en el que ya se incluyen algunos servicios extras.

5.3.2. Las opciones de acceso de un usuario avanzado.

Si el usuario tiene mayores privilegios, por ejemplo, ha pagado el acceso a Internet, se puede observar algunas variaciones en el sistema. A la descripción que se comentó en el punto 5.3.1, como se puede observar en la Figura 5.5, aparece una nueva opción de acceso a Internet. Esta opción permitirá al usuario acceder a Internet. Para mayor información de lo que ocurre, ir a la sección 5.4 en la que se puede encontrar una explicación detallada.

El acceso a Internet, también queda señalado en el menú superior. Se puede observar que el nivel de acceso a cambiado de "básico" a "completo".

5.3.3. Las opciones de un administrador.

Un administrador del sistema, como es de esperar, tiene algunas opciones adicionales que no poseen los usuarios clásicos del sistema. En la Figura 5.6 se puede observar el acceso de un administrador al sistema y las modificaciones en el menú que aparecen.

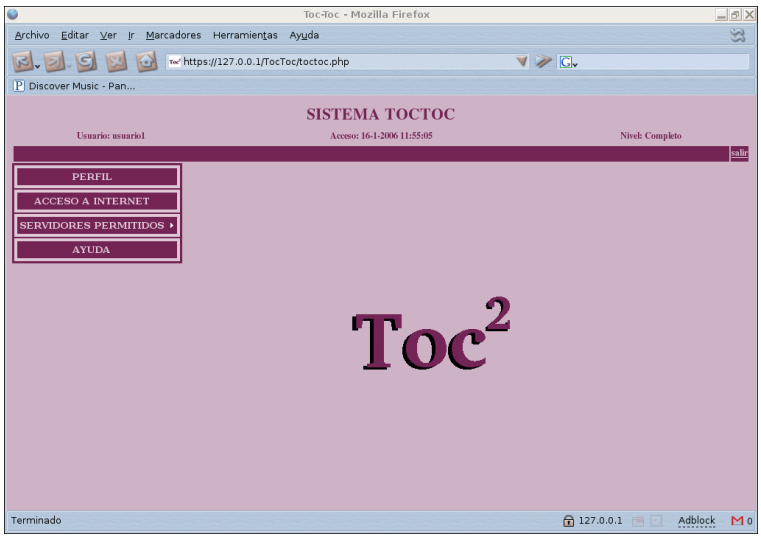

Figura 5.5: El sistema visto por un cliente con mayor acceso.

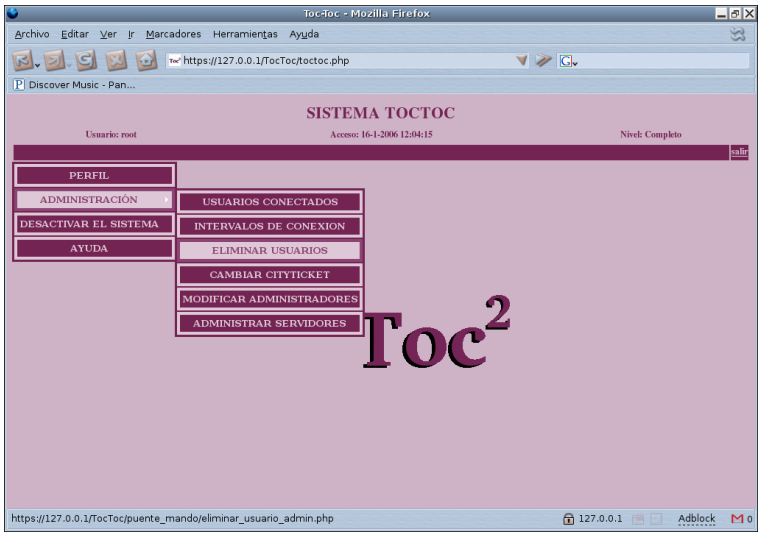

Figura 5.6: Las opciones de un administrador.

Por un lado aparece una opción de administración que si se selecciona, da paso a un nuevo menú desplegable con todas las opciones de administración listadas. Desde aquí se puede controlar los usuarios del sistema y obtener algunos datos de los mismos a partir del *log* del sistema, así como añadir o quitar servidores gratuitos y permitir el acceso a Internet a cualquier usuario. Todo esto quedará explicado en el apartado 5.5 de este capítulo.

Por otro lado, aparece una nueva opción que será "Desactivar el sistema" o "Activar el sistema" según el estado actual del mismo. Esta opción permite

activar (o desactivar) el sistema de forma remota a un administrador que tenga acceso al mismo.

5.3.4. Algunas variaciones dependiendo de la máquina.

Aparte del tipo de usuario, lo anteriormente indicado tiene una excepción adicional. Si en vez de darse de alta en un cliente de la red, un usuario se da de alta en la propia máquina que contiene el sistema, algunas opciones desaparecerán ya que carecen de significado. Por ejemplo, la opción de acceso a Internet desaparece, ya que esta opción solamente controla el acceso de las máquinas situadas en la red *wireless* con el exterior y la máquina local ya tiene acceso directo a Internet.

Otro ejemplo de esto mismo, serían los servidores permitidos, que desaparecen por el mismo motivo anteriormente citado.

5.4. Acceso a Internet.

Un usuario con los privilegios necesarios, puede acceder a Internet de una forma sencilla. Una vez dado de alta en el sistema y habiendo introducido su nombre de usuario y su contraseña, podrá seleccionar la opción "Internet" en el menú de opciones del sistema. Seleccionar esta opción, llevará a la página que se muestra en la Figura 5.7. Como se puede observar, en el menú superior aparecen algunos datos de la conexión, aparte del nombre de usuario. Estos datos son, la IP del cliente en la red local del sistema, y la velocidad de bajada y subida que tiene contratados. La velocidad de conexión depende del *Cityticket* que tenga contratado.

Una explicación inferior indica que, si tiene algún navegador que impida el acceso a las ventanas flotantes, debe de dar permiso a las provenientes de este portal o el acceso a Internet no funcionará correctamente.

Figura 5.7: Acceso a Internet.

Una vez aparece la ventana flotante, el sistema desparecerá y dará paso a la página web que el usuario escribió en un principio. La ventana flotante mantiene la conexión a Internet. Por ello, debe de estar abierta, aunque puede permanecer minimizada. Si la ventana se cierra mediante el enlace "Cerrar", automáticamente se cancelará la conexión a Internet. Si se cierra la ventana de algún otro modo, como por ejemplo mediante el botón con el aspa que aparece en la esquina superior derecha de la ventana, la conexión desaparecerá en unos minutos, dependiendo del margen que se haya dado en la configuración del sistema. Esto ocurre, ya que esta ventana es la que avisa al servidor que el cliente todavía está activo, y cerrarla equivale a indicar que el cliente ha sido apagado.

Como se puede observar en la ventana, aparte del aviso y del enlace para cerrarla, aparece un contador de tiempo que indica al cliente el tiempo que lleva conectado a Internet. Esto es muy útil, ya que el sistema puede tener un sistema de tarificación por tiempo real en Internet (como un teléfono) o un sistema de tarificación por días (lo que normalmente se llama "tarifa plana"). Por supuesto, este tiempo quedará registrado en el archivo de *logs* del sistema.

5.5. Opciones de administración.

Si el usuario que accede al portal tiene algún derecho de administración, este tendrá algunas opciones adicionales que le permitirá controlar el sistema.

5.5.1. Usuarios conectados.

Esta opción permite a un administrador visualizar un listado de todos los usuarios que están accediendo a Internet a través del sistema y la fecha en la que realizaron esa conexión. Por ejemplo, un resultado de esto se puede observar en la Figura 5.8.

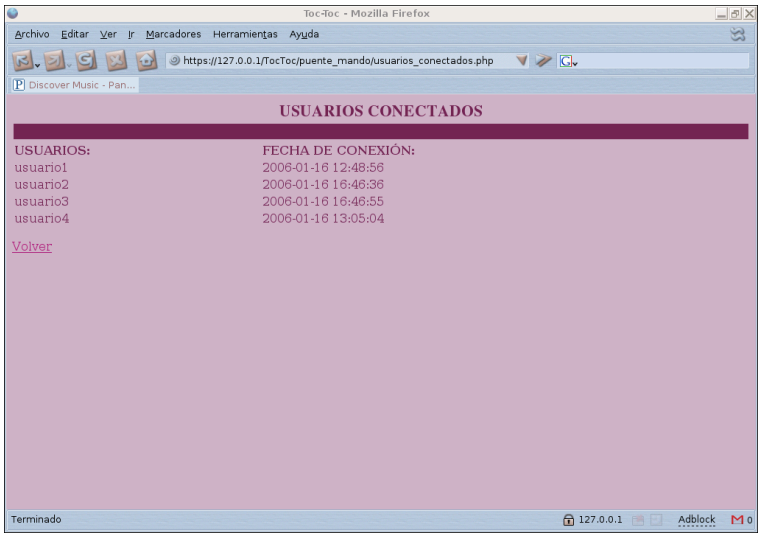

Figura 5.8: Usuarios conectados.

Esta opción es meramente informativa y no se puede realizar ninguna acción más.

5.5.2. Intervalos de conexión.

En algunas ocasiones, puede ser importante saber en que momentos un usuario en concreto se ha conectado. Saber de que horas a que horas a accedido a Internet y cuanto tiempo en total lleva consumido. Esto se puede consultar mediante esta opción. Un pequeño formulario permite introducir el usuario que se quiere consultar, para después mostrar un listado de IPs y fechas según se ha ido conectando el usuario a lo largo del sistema.

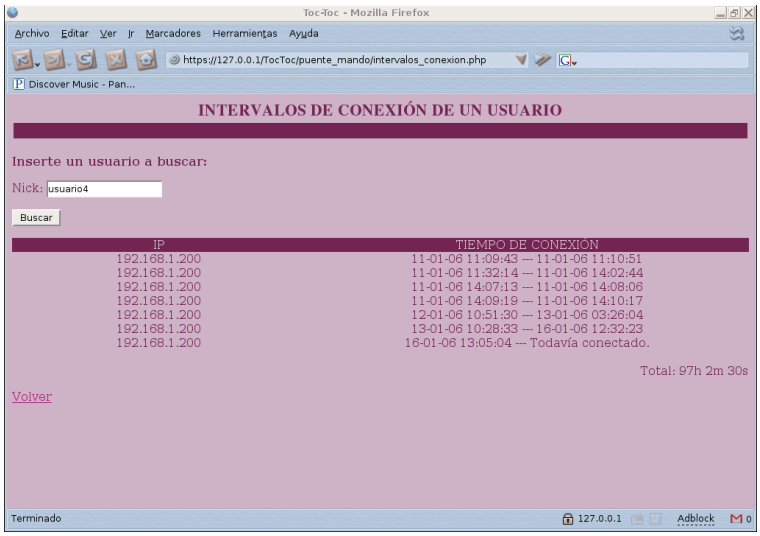

Figura 5.9: Intervalos de conexión.

Como se puede observar en la Figura 5.9, indica explícitamente si el usuario todavía sigue conectado al sistema y cuanto tiempo en total lleva conectado en el mismo.

5.5.3. Eliminar usuarios.

En todo sistema siempre tiene que existir alguna forma de eliminación de usuarios. Ya sea por que se ha dado de baja en el sistema o por que se ha convertido en un usuario no deseado, un administrador puede borrar al usuario. Si se selecciona esta opción aparece un formulario en el que si se introduce el nombre de usuario, este se borra. Para facilitar las cosas al administrador, un listado con todos los usuarios se presenta poco más abajo, de manera que seleccionando uno de ellos, lo escribe directamente y lo deja listo para borrar.

Hay que señalar que no existe confirmación en la eliminación de un usuario, por lo que si se pulsa el botón pertinente, el usuario escrito será eliminado sin posibilidad de retornar al estado anterior.

Otro detalle importante, es que existe un usuario por defecto en el sistema que se le ha dado el nombre de *root*. Este usuario es el único que no aparece en el

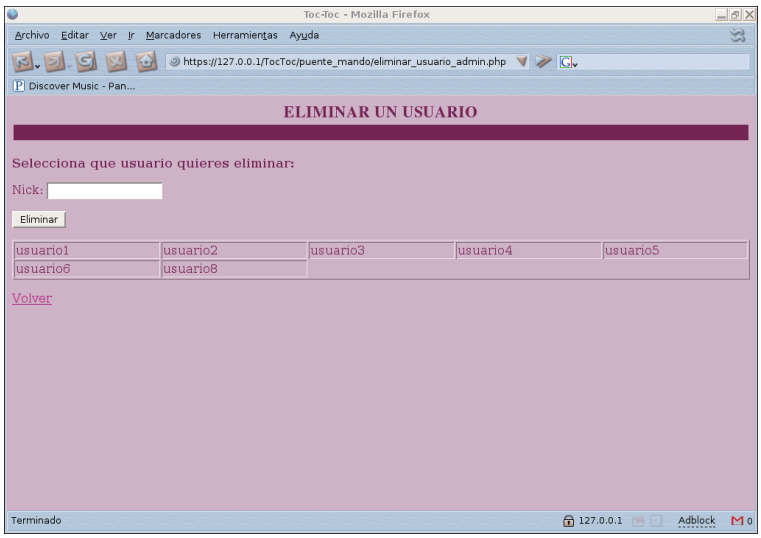

Figura 5.10: Eliminar usuarios.

listado ya que se considera que es inherente al sistema y permite al administrador tener siempre una cuenta de acceso.

5.5.4. Cambiar *Cityticket*.

Como se ha señalado en la sección 5.11, el *Cityticket* representa el nivel de acceso a Internet al que puede optar un usuario. El valor de cada nivel se define en la configuración del sistema, cuando se instala. Por defecto existen diez niveles, del uno al nueve. Tienen un valor creciente de velocidad de conexión. Así el primer valor representa que el cliente no tiene conexión a Internet, y el valor noveno representa que tiene una velocidad de bajada de 2 MB/s y 1 MB/s de subida. El valor número diez se guarda para que sea el propio administrador el que decida la velocidad de forma particular a cada usuario.

El siguiente campo a rellenar es una fecha en formato año (con cuatro dígitos numéricos) - mes (con dos dígitos numéricos) - día (con otros dos dígitos). Este campo indicará al sistema cuando caduca el *Cityticket* de forma que el usuario deje de tener conexión a Internet cuando se llegue a esa fecha determinada.

Como se puede observar en la Figura 5.11, a *usuario6* se le esta dando una conexión personalizada por el administrador. El resto de usuarios tienen el campo de velocidad desactivado pues tienen un *Cityticket* predeterminado.

5.5.5. Modificar administradores.

Cuando un sistema crece rápidamente, puede resultar engorroso que una única persona se encargue del mantenimiento. Para ello, existe una opción para crear nuevos administradores en el sistema. En TocToc se han diferenciado dos tipos de administradores: los administradores propiamente dicho y los moderadores. Los administradores son capaces de realizar todo lo comentado en este capítulo, mientras que los moderadores son unos administradores a los que se

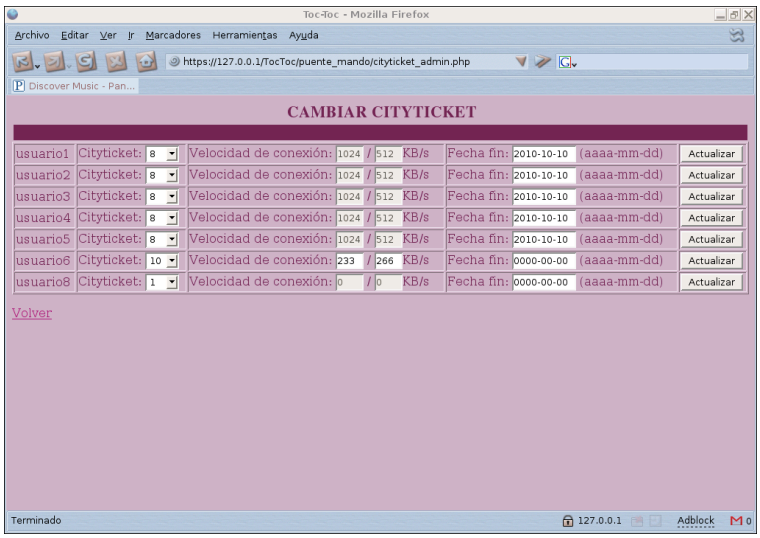

Figura 5.11: Cambiar Cityticket.

les ha reducido su función. Por ejemplo, no pueden apagar el sistema o crear nuevos administradores. La idea de que existan moderadores, es principalmente, un usuario que se encargue exclusivamente de otorgar conexiones a Internet a los nuevos clientes, y se desentienda de la configuración del sistema.

Figura 5.12: Creación de administradores.

Como se ve en la Figura 5.12, para añadir un moderador o administrador, este tiene que ser dado previamente de alta como usuario. Esto es debido a que lo que realmente se hace es interactuar con el listado de usuarios del sistema.

Estos usuarios son presentados en orden alfabético para facilitar su localización.

5.5.6. Administrar servidores.

Esta opción se refiere a gestionar aquellos administradores a los que el sistema permita un acceso gratuito independientemente del nivel de acceso a Internet de los clientes. Existen dos formas de dar acceso a los servidores de forma gratuita. Mediante puertos y por IPs. La diferencia entre una forma u otra está explicada en la sección 4.7.

Los distintos campos a rellenar por un administrador para la administración de los mismos son (ver Figura 5.13):

1. El nombre del servidor: que será el nombre que aparezca en el menú de servidores permitidos (obligatorio).
2. El puerto local: si se rellena este campo, se indica que se quiere un servidor de acceso gratuito mediante puertos. Si se deja en blanco, el sistema entenderá que el servidor será libre mediante IPs.
3. La dirección web: se escribe la *URL* del servidor al que se quiere dar acceso (obligatorio si no se escribe el campo de IP).
4. La dirección IP: si se rellena este campo, se accederá al servidor mediante esta IP. Sino, el servidor resolverá la dirección y apuntará automáticamente la IP obtenida.
5. Puerto destino: indica el puerto de escucha del servidor destino. Si se dejan en blanco, se asignará por defecto el puerto 80 para conexiones normales, y el 443 para conexiones seguras.
6. Borrar: si se activa esta casilla, se borrará el servidor seleccionado.

Conviene recordar que, como en otras páginas, el listado de servidores aparece en orden alfabético según el nombre con el que se dan de alta.

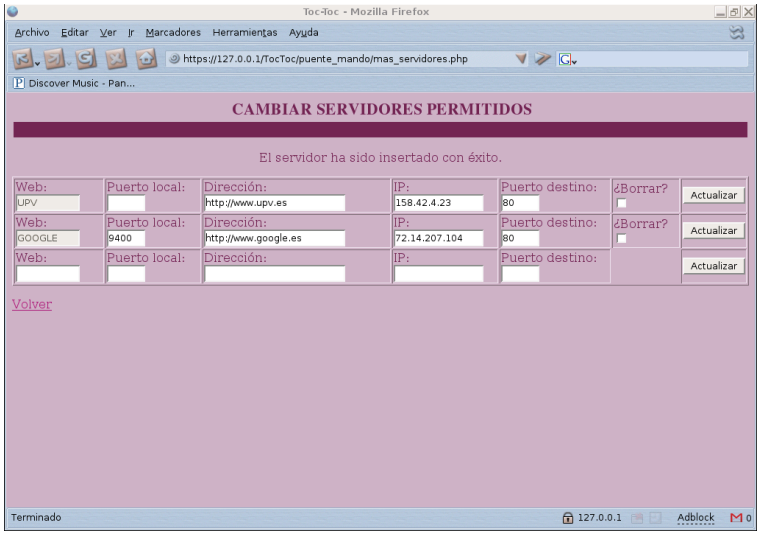

Figura 5.13: Servidores permitidos.

Capítulo 6

Trabajos relacionados.

Una pregunta que queda por resolver, es la de "¿por qué no se usa una herramienta ya existente?". Como respuesta a la misma se puede decir que, al principio de la generación del sistema, se realizaron distintas pruebas de adaptación de algunos *Captive Portals* existentes en el mercado ([8, 10, 14, 2]) y así intentar ahorrar el esfuerzo que supone generar uno propio. Sin embargo se ha decidió descartar la idea por los motivos que se explican a continuación.

6.1. Problemas con *NoCat*.

El principal problema que se obtuvo con *NoCat*[1]es el propio funcionamiento del mismo. Para este sistema, lo que se desea desde un principio es la generación de un portal con distintos servicios particulares públicos, a los que pueda acceder cualquiera, previa cumplimentación de un formulario. El acceso a Internet viene dado a posteriori, si dicho cliente así lo solicita y si este tiene derechos para el mismo.

Por contra, *NoCat* lo que realmente hace, es un sistema de acceso en el que si el cliente está dado de alta, puede acceder a todos los servicios del mismo, y si no es así, no puede acceder a ninguno. No existe una situación intermedia que permita el acceso a algunos servicios propios como los ofertados en el ejemplo 8.1 y necesites un nivel de acceso superior para acceder a otros servicios.

Se evaluó la opción de realizar un primer paso para la red local, mediante un portal con todos los servicios deseados, y luego regular el acceso a Internet mediante una redirección al *NoCat*. Sin embargo, la propia estructura de *NoCat*, precisa que la *Splash Page* sea el propio *NoCat*, ya que su funcionamiento requiere unos datos que envía el navegador del cliente mediante la primera petición de acceso a Internet por parte del usuario. Si esto se cambia por una redirección desde el propio portal, *NoCat* acaba confundiendo la máquina que aloja al portal como la máquina cliente, dando acceso a esta en vez del cliente.

Un último motivo para desechar la idea de *NoCat* es que este, controla la velocidad de conexión mediante tres grupos de usuarios que se pueden dividir en

[1]Por *NoCat*, nos referiremos exactamente a la versión *NoCat-Auth-0.82* y *NoCatAuth-nightly*. Ya que ambos presentaron los mismos problemas. El programa *NoCat-Splash*, aunque también es un *Captive Portal*, no es un sistema de control de acceso a Internet mediante autentificación de usuarios, por lo que no resulta de interés para este trabajo. Para más información, consultar [8].

lentos, medios o rápidos, mientras que no permite una conexión personalizada para cada usuario. A veces es interesante tener un sistema que permita controlar de forma individualizada cada usuario.

6.2. Problemas con *Firstspot.*

La primero que hay que tener en cuenta con *Firstspot* [10], es que esta dentro de lo denominado como Software propietario, es decir, que no se proporciona el código fuente, por tanto no se puede modificar nada del sistema original que te proporciona la empresa que desarrolló el software. Las ventajas que ofrece este sistema frente a *NoCat* son que si que permite acceder a un listado de servidores permitidos, de forma previa a la identificación del usuario, permitiendo así el acceso a algunos servicios de forma gratuita. También tiene una mayor facilidad de gestión, de configuración y alguna opción propia extra como el control de clientes mediante IPs.

Como desventajas se puede señalar, que tiene varios defectos en el funcionamiento del mismo. A la hora de configurar los clientes permitidos, con la versión actual (versión 3), tiene defectos a la hora de almacenar datos y recuperarlos, perdiendo datos relevantes del cliente. Otros defectos relevantes, son que al igual que *NoCat*, tampoco permite la compenetración con un sistema externo al mismo, por lo que no se puede complementar el sistema con servicios propios extra.

6.3. Problemas con algunas otras alternativas.

Aparte de los dos ejemplos concretos explicados anteriormente, existen unas tendencias en los *Captive Portals* (como el *m0n0wall*) que consiste en la creación de un CD de arranque con todo lo necesario del mismo. La configuración del mismo se guarda en un disquete que tendrá que estar siempre insertado en la disquetera. La ventaja de estos sistemas es que carecen de sistema operativo por lo que permite a las máquinas poco potentes, orientar todos sus esfuerzos a la ejecución exclusiva del *Captive Portals* y pasar un equipo inservible a un punto de acceso potente. Por supuesto, la desventaja más clara es que el no disponer de un sistema operativo, impide la compenetración del mismo con cualquier otro servicio que se quiera ofertar, como por ejemplo, un servidor Apache, así como el uso de tarjetas de red que no estén contempladas en el propio *Captive Portal.*

Capítulo 7

Experimentación.

Este capítulo pretende mostrar el comportamiento del sistema TocToc cuando tiene un ancho de banda limitado y los clientes piden más ancho de banda que el que puede proporcionar. Lo que se quiere comprobar es como un nuevo cliente, al entrar en un sistema saturado y solicitar su ancho de banda correspondiente, interfiere con el ancho de banda de los demás clientes y las consecuencias que esto conlleva. Mostrar si este cliente logra obtener algo de ancho de banda en su intento de acceso, y si este es proporcional de alguna manera, con el ancho de banda del resto de clientes y la relación existente entre el ancho de banda contratado. Es importante observar, si en un medio con una capacidad inferior a las necesidades de los clientes, estos tienen un ancho de banda similar entre ellos, independientemente del solicitado por cada uno o por contra, todos reducen su ancho de banda de manera proporcional de manera que mantenga algún privilegio aquel cliente que tenga algún ancho de banda mayor.

7.1. Dispositivos utilizados en las pruebas.

Para el estudio del sistema, se ha utilizado un ordenador que actúa como servidor. Dicho computador dispone de dos tarjetas de red, una con salida a Internet y la otra conectada a un punto de acceso. El ordenador utilizado es un *AMD XP* 1800+ con 512 MBs de *RAM DDR*, aunque puede ser utilizado uno de menor capacidad sin problemas siempre y cuando pueda albergar un Linux. El sistema operativo utilizado ha sido un Fedora Core 3 y después se ha exportado a una Debian 3.1 para comprobar su exportabilidad, con idénticos resultados.

La tarjeta *Ethernet* con salida a Internet (que puede ser cualquier otro dispositivo de conexión como un módem *ADSL*) tiene que tener una conexión suficiente para dar cabida al tráfico de todos los clientes que se pretendan alojar en el sistema.

Las pruebas de cliente han sido realizadas en varios portátiles con distintos sistemas operativos (tanto Windows XP, como Fedora Core 3 y SuSE Linux). Cada portátil tiene que tener una tarjeta de red inalámbrica de distintas marcas para poder conectarse al servidor. Los modelos oscilan entre distintas épocas, siendo el más lento un Celeron a 150 MHz con 96 MB de RAM y el mayor un Pentium IV 1700 MHz con 512 de RAM.

7.2. Forma de obtener resultados.

La estructura de trabajo consiste en montar el sistema y distribuir los distintos equipos alrededor del mismo. Un equipo central esta conectado a un punto de acceso que es el que genera la red inalámbrica. Un segundo equipo contiene un servidor FTP exclusivo para realizar la descarga de diversos ficheros de distinto tamaño sin interferencias entre otras peticiones externas. En este servidor se puede regular el ancho de banda que sirve a los clientes para simular distintas situaciones con posibles limitaciones de ancho de banda.

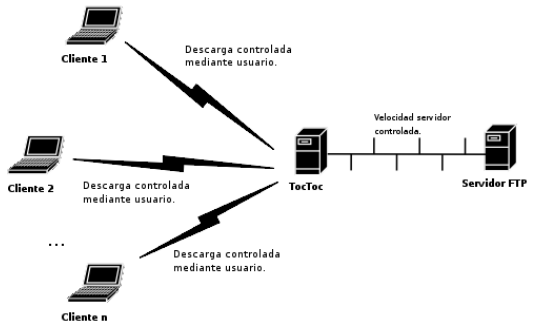

Figura 7.1: Estructura de pruebas en el sistema.

La metodología seguida para obtener los resultados de las pruebas consiste en activar un *script* en el ordenador central que despierta a los distintos clientes pasado un tiempo preestablecido. Con este margen de tiempo, se puede observar las consecuencias de la entrada de un cliente en el sistema. Cada uno de estos clientes, que ya se han dado de alta en el sistema TocToc con un usuario con acceso a Internet, ejecuta un *script* que lanza una petición FTP al servidor que simula ser Internet. Por ejemplo, podemos hacer que dos equipos accedan al servidor con una diferencia de 30 segundos y se descarguen un fichero de 10 MB si configuramos ambos *scripts* adecuadamente. En el Algoritmo 10 se muestra la parte que se ejecuta en el ordenador que contiene al sistema TocToc y la parte que se ejecuta en cada uno de los clientes.

Todas las peticiones y paquetes pasan por el ordenador central, que a su vez tiene activado la herramienta *tcpdump* que registra el tamaño, procedencia y destino de cada paquete en un fichero (llamado, en nuestro caso, *traza.dump*). Estos pasos quedan resumidos en la Figura 7.2.

Una vez se obtiene el archivo con todo el registro de paquetes que han pasado por el servidor, la metodología que se ha seguido ha sido la siguiente:

1. Se descompone el archivo traza.dump en un archivo por cliente que contiene únicamente los paquetes que maneja ese cliente. Esto lo realiza el *script* "adapta_datos.sh".

2. Cada uno de estos archivos creados, se le pasa al Octave, que es un lenguaje de alto nivel para computación numérica [**?**], de manera que éste, mediante las funciones "archivo_vector", "agrupa_intervalo", "fusiona_matrices" y "genera_graficas" creadas para este propósito determinado; lee cada archivo que se le indica, lo transforma en un vector que contiene los paquetes

Algorithm 10 Forma de ordenar la descarga de un archivo a distintos clientes.

```
### MAQUINA PRINCIPAL: Archivo ssh.sh
$TIEMPO=30
for $Host in 192.168.1.195 192.168.1.196 192.168.1.197 192.168.1.198
do
  ssh $Host ftp.sh 10M
sleep $TIEMPO
done

### MAQUINA CLIENTE: Archivo ftp.sh
echo "
open $HOST_FTP
user $USUARIO_FTP $PASSWORD_FTP
get $1
" | ftp -in
```

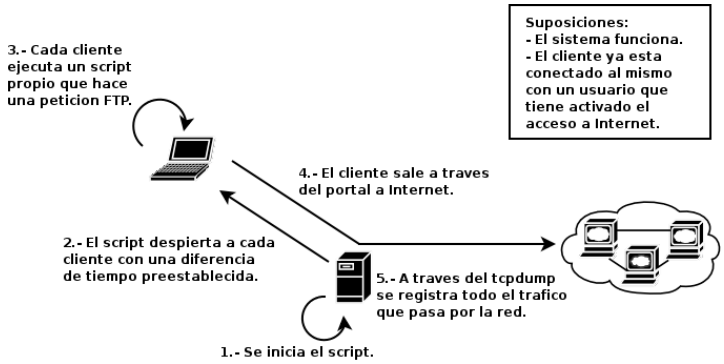

Figura 7.2: Forma de obtener los datos.

IP del cliente relacionados con el tiempo y los agrupa en intervalos de muestras (por ejemplo, contabiliza los paquetes cada segundo o cada 10 segundos). La función "fusiona_matrices" desarrolla una matriz en la que aparecen la relación de tiempo, con los paquetes de cada uno de los clientes separados en columnas. Por último, esta matriz se guarda en un archivo llamado "grafica.txt". Para más información sobre el funcionamiento sobre el Octave, dirigirse a [**?**].

3. Por último, entra en acción el *script* "obtiene_grafica.sh" que ejecuta el Gnuplot y obtiene la gráfica con los datos determinados.

Esto se visualiza en la Figura 7.3. Ahí se aprecia la secuencia de pasos y los archivos que se generan después de cada uno de ellos.

Estos pasos se repiten para cada una de las gráficas, es decir, para cada cambio de la velocidad de los clientes o para cada cambio en el límite de velocidad del servidor.

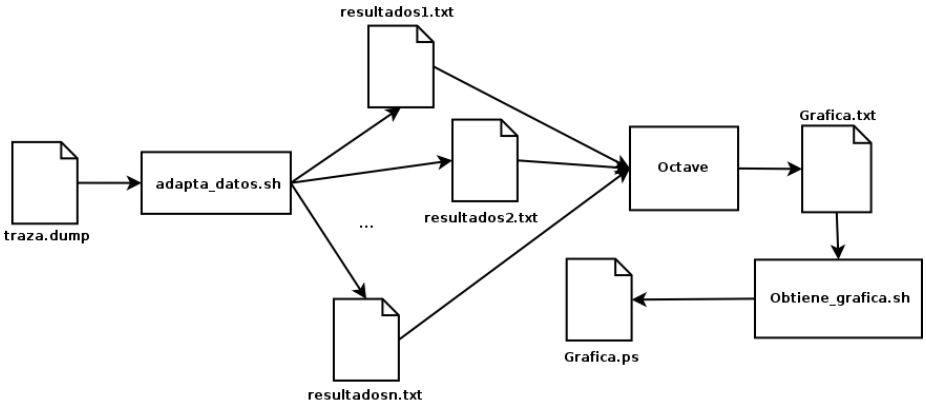

Figura 7.3: Paso de datos a gráficas.

7.3. Gráficas y resultados obtenidos.

A continuación se presentan los resultados obtenidos con la experimentación. Los clasificaremos según el número de clientes que participan en las pruebas y el tiempo de separación entre la petición que lanza el primer cliente, y la del segundo.

7.3.1. Comprobación del funcionamiento para un único cliente.

Lo primero que hay que hacer es probar el funcionamiento con un único cliente. La gráfica 7.4 muestra como se comporta un único cliente a distintas velocidades. En esa gráfica, el eje X marca el tamaño de los distintos archivos, mientras que el eje Y marca el tiempo necesario para descargar dicho archivo. Cada función representa una velocidad con la que se ha probado el mismo cliente.

Como se puede observar las funciones son lineales, directamente proporcional al tamaño del archivo e inversamente proporcional a la velocidad del mismo. Si se comparan unas funciones con otras, se puede observar que son proporcionales entre sí de acuerdo con la velocidad que representan.

La siguiente gráfica, la número 7.5, representa la oscilación de velocidad que se presentan al bajar distintos archivos con un mismo cliente. Para la generación de estas pruebas, se han generado archivos de distinto tamaño (1, 2, 3, 5, 8, 10, 12, 15, 20, 25, 30, 40 50, 80 y 100 Megas). Se ha medido el tiempo de descarga de cada uno y se ha calculado la velocidad media. Se ha probado 6 veces con el mismo cliente a distinta velocidad (16, 32, 64, 128, 512, 1024 KB/s) que representa los seis valores del eje X. El eje Y representa la velocidad real obtenida. Cada intervalo resultado, esta compuesto por el valor mínimo, el valor máximo y la media obtenida.

Para poder observar detalladamente la variación de los intervalos, se adjunta una tabla (tabla 7.1) con el valor numérico de cada uno.

Como se puede observar, los valores se ajustan bastante a los deseados. La única excepción es la prueba 6, que es bastante inferior al esperado. Esto no es de extrañar, sabiendo que 1024 KB/s son 8192 Kb/s y esto se acerca bastante

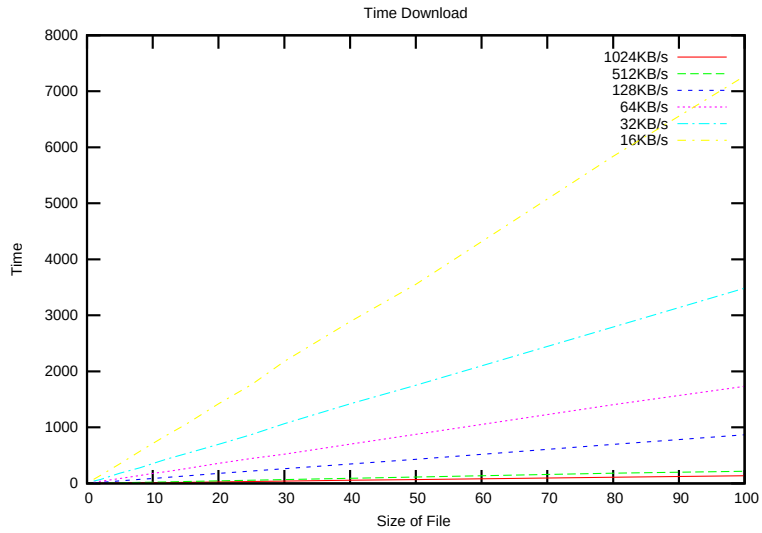

Figura 7.4: Relación de velocidad y tamaño de archivo para un único cliente.

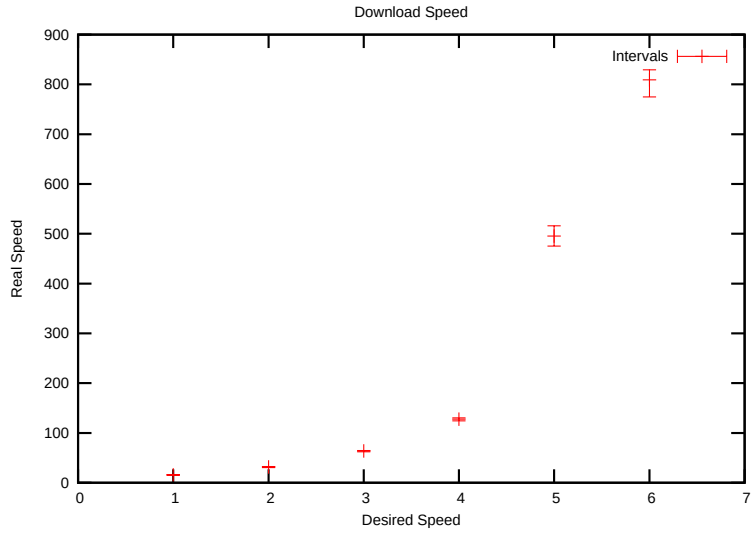

Figura 7.5: Intervalos de velocidad.

al límite que puede soportar la red *wireless* utilizada, que para este ejemplo es de 11 Mb/s.

7.3.2. Pruebas realizadas con dos clientes.

Una vez ya se sabe que el comportamiento del sistema es bueno para un único cliente, hay que comprobar la reacción del mismo cuando coexisten dos clientes a la vez. Para ello, se van a probar distintas velocidades entre clientes

Valor X	Velocidad Deseada	Velocidad Mínima	Velocidad Máxima	Velocidad Media
1	16 KB/s	15.7950 KB/s	16.0300 KB/s	15.4300 KB/s
2	32 KB/s	31.9171 KB/s	32.3800 KB/s	30.5800 KB/s
3	64 KB/s	63.8150 KB/s	64.9200 KB/s	62.1800 KB/s
4	128 KB/s	127.7221 KB/s	130.3100 KB/s	124.1000 KB/s
5	512 KB/s	495.3293 KB/s	515.9300 KB/s	475.4400 KB/s
6	1024 KB/s	809.3464 KB/s	829.2400 KB/s	775.0300 KB/s

Cuadro 7.1: Intervalos de velocidad.

y se van a seleccionar límites al servidor.

Velocidad de ambos clientes a 16 KB/s y velocidad de descarga máxima a 512 KB/s. La primera gráfica (Figura 7.6) como resultado de la interacción entre dos clientes, pretende demostrar que, dos clientes en un mismo sistema que tiene capacidad para albergar a ambos, tienen un comportamiento correcto. Para ello se ha solicitado una descarga de un archivo de 50 MB.

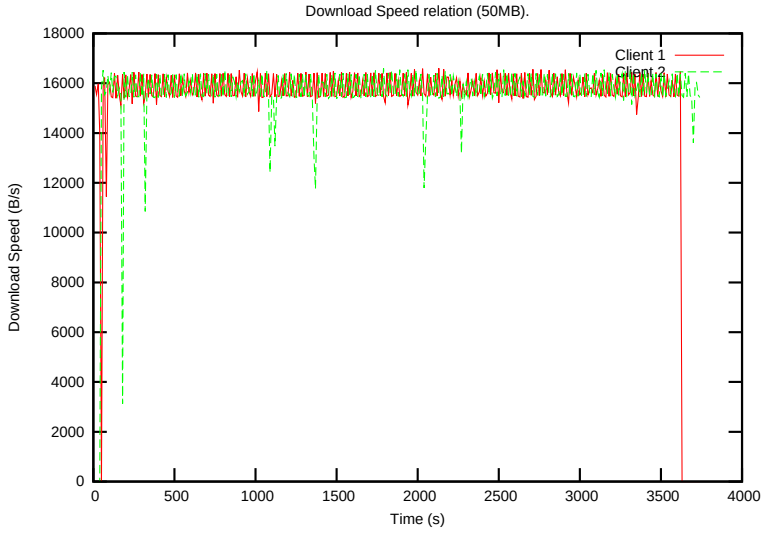

Figura 7.6: Dos clientes con 16 KB/s.

Como se puede observar, ambos clientes se mantienen en la línea de los 16 KB/s, con algunas oscilaciones esporádicas, pero que nada tienen que ver con la presencia del otro cliente, sino que son debidas al funcionamiento de la propia red *Wireless*.

Velocidad de ambos clientes a 16 KB/s y velocidad de descarga máxima a 16 KB/s. En este caso, se quiere comprobar la reacción del sistema al entrar dos clientes que, en principio, tienen los mismos privilegios. Si dos clientes con igual velocidad entran en un sistema que no tolera esta velocidad, deberá de equilibrar la carga de ambos clientes de una forma mas o menos equitativa. Para

este ejemplo, se ha seleccionado un archivo de tamaño 10 MB para conseguir un tiempo de prueba algo menor.

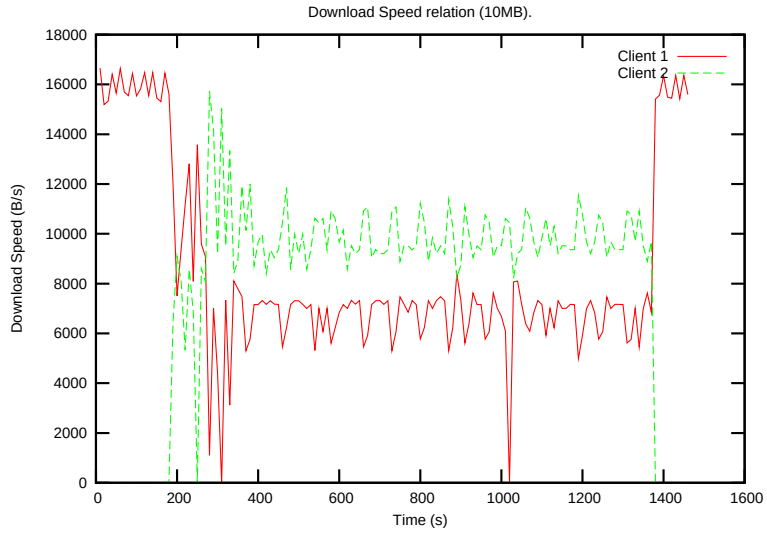

Figura 7.7: Dos clientes con 16 KB/s en un medio restringido.

Como se puede observar en la Figura 7.7, con la entrada del segundo cliente en el instante de tiempo 180, el primero cede parte de su ancho de banda a este. En este ejemplo, el segundo cliente recibe algo más de ancho de banda que el primero durante más tiempo, consiguiendo acabar algunos segundos antes que este. Esto es debido a que el reparto de recursos no siempre es homogéneo en una red TCP/IP de manera que puede ocurrir que un cliente acapare algunos recursos de más. Podemos ver un ejemplo en la siguiente sección, en donde la disputa por los recursos sea un poco más agresiva.

Velocidad de ambos clientes a 512 KB/s y velocidad de descarga máxima a 16 KB/s. Si probamos el sistema con dos clientes que solicitan mucho más ancho de banda que el permitido, se puede observar una oscilación entre ellos un poco mayor. Sin embargo, como se aprecia, cuando uno de ellos consigue durante unos instantes una mayor ancho de banda, lo pierde el otro cliente, ya que la suma de ambas velocidades nunca puede ser superior a los 16 KB/s permitidos.

En la Figura 7.8, se puede ver como a pesar de todo, la media aproximada de cada cliente se mantiene sobre los 8 KB/s que sería lo que les correspondería a cada uno de ellos.

Saturación debido a las limitaciones de la *Wireless*. La red *Wireless* utilizada en pruebas funcionando correctamente, tiene un ancho de banda ideal de 11 Mb/s. Sin embargo, rara vez se consigue alcanzar este límite realmente. En las pruebas realizadas, esto se ve claramente cuando dos clientes con un derecho de ancho de banda elevado (1024 KB/s) y a su vez el servidor tiene su límite puesto también a esa velocidad. Los resultados a esperar es que se llegara

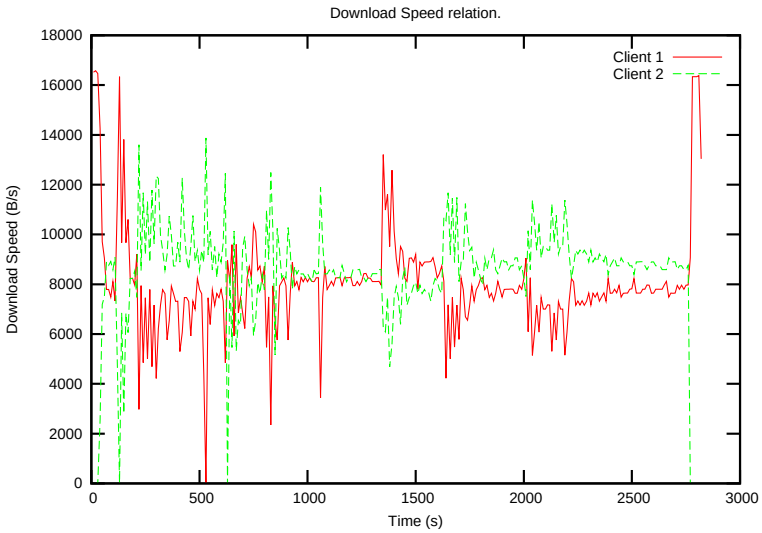

Figura 7.8: Dos clientes con 512 KB/s en un medio restringido.

a la velocidad de 1024 KB/S (8 MB/s) que se supone que es inferior al ancho de banda máximo de la red. Sin embargo, como se muestra en la Figura 7.9, nunca se llega al valor máximo, sino que la media oscila sobre los 600 KB/s (unos 4.8 MB/s).

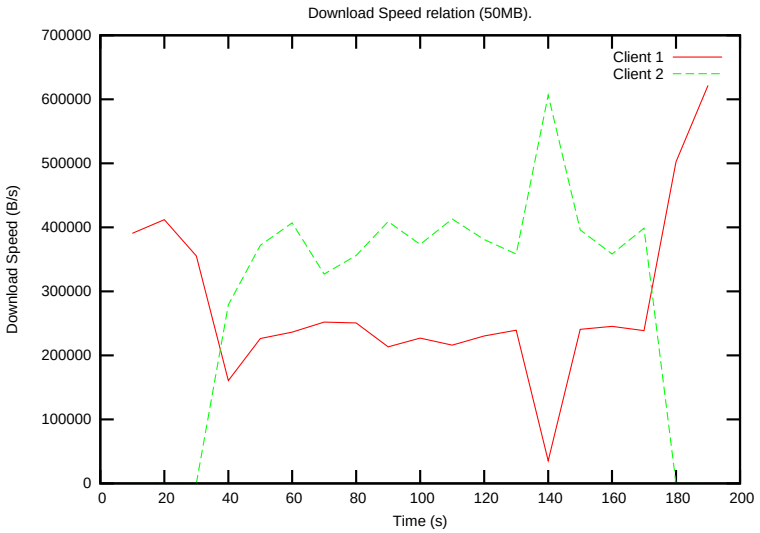

Figura 7.9: Limitaciones de velocidad debido al punto de acceso.

7.3.3. Pruebas realizadas con cuatro clientes.

Una vez se ha comprobado la reacción del sistema ante dos clientes, conviene repetir las pruebas para un número superior. De esta forma se asegura que el funcionamiento del sistema siga siendo el correcto.

Cuatro clientes a 16 KB/s y un límite de ancho de banda muy superior al necesario. Como en el caso de los dos clientes, se comprueba que el funcionamiento del sistema con estos cuatro clientes sea el correcto. Para ello, lo primero es ver si el funcionamiento del sistema cuando la red no se satura.

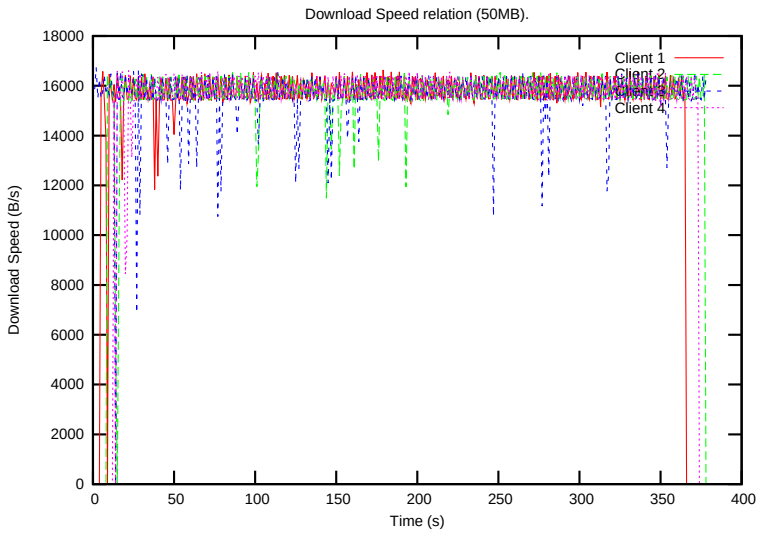

Figura 7.10: Cuatro clientes sin límites en la red.

Como se observa en la Figura 7.10, según van entrando los clientes en el sistema, van ocupando su ancho de banda correspondiente. De forma que una vez están los cuatro en el mismo, tienen todos sus 16 KB/s permitidos, salvo algunas oscilaciones temporales en el sistema, probablemente causadas por el mantenimiento de la conexión *Wireless*.

Cuatro clientes a 16 KB/s y un límite total de 32 KB/s. En este ejemplo, se puede observar el comportamiento del sistema con un ancho de banda limitado, que solamente puede albergar a dos clientes, mientras que entran cuatro en el sistema. Como se observa en la Figura 7.11, la entrada de los dos primeros clientes al sistema no produce perturbación ninguna, mientras que la entrada de un tercer cliente, obliga a disminuir la velocidad a los dos anteriores. Una vez están activo los cuatro, la media se sitúa en los 8 KB/s, aunque se puede observar una clara competición por los recursos que provocan oscilaciones en las velocidades de los cuatro.

Según acaban los clientes en la figura citada anteriormente, se puede observar como la velocidad de los demás clientes van llegando a su máximo permitido, para estabilizarse sobre el segundo 120, en el que ya se sitúan en los 16 KB/s correspondientes.

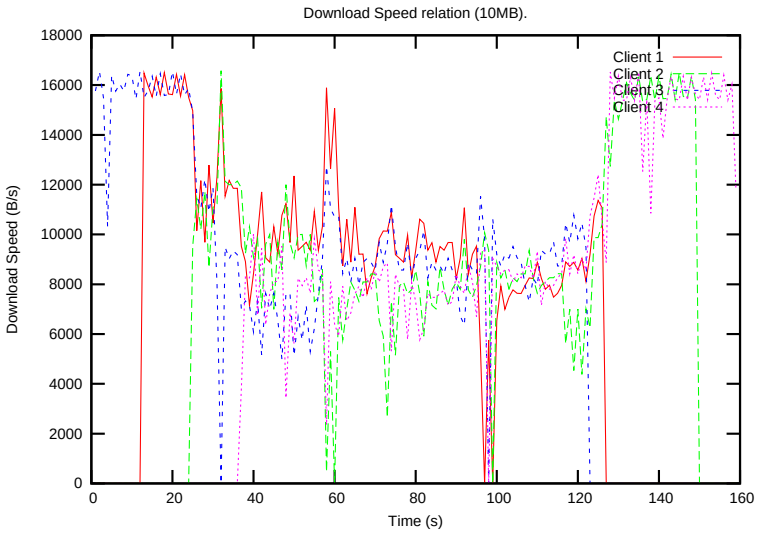

Figura 7.11: Cuatro clientes con 16 KB/s y un límite de 32 KB/s.

Otros casos estables, cuatro clientes a 64 KB/s y diversos límites de descarga. Para comprobar el comportamiento del sistema ante una diversidad de clientes con velocidades homogéneas, se han realizad más pruebas con diversas velocidades. Una de ellas, es dotar a cuatro clientes de un ancho de banda de 64 KB/s, y realizar pruebas con distintas velocidades máximas. Se busca en estos ejemplo, realizar un número mayor de pruebas para analizar el sistema antes de sacar conclusiones.

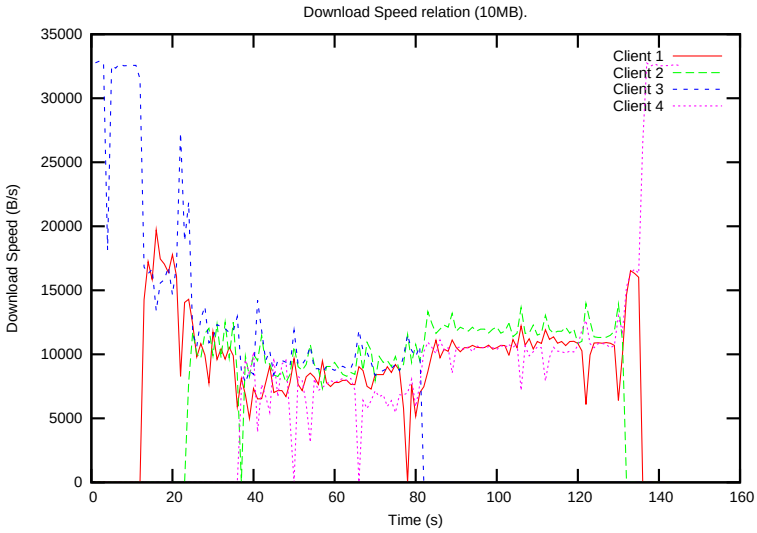

Figura 7.12: Cuatro clientes a 64 KB/s y un ancho de banda de 32 KB/s.

Como se puede observar en la Figura 7.12, el comportamiento de los clientes vuelve a ser equitativo cuando los recursos son limitados y los clientes tienen las mismas condiciones. En esta gráfica se observa, que aún cuando el primer cliente satura la red, la entrada sucesiva de otros clientes obliga a un reparto equitativo de los recursos. Esta situación se mantiene en las distintas pruebas realizadas, en las que el servidor va aumentando el ancho de banda total disponible hacia los clientes.

Una vez se ha incrementando el ancho de banda lo suficiente, se llega al resultado de la Figura 7.13, en el que cada uno de los clientes obtiene el ancho de banda que le ha sido otorgado.

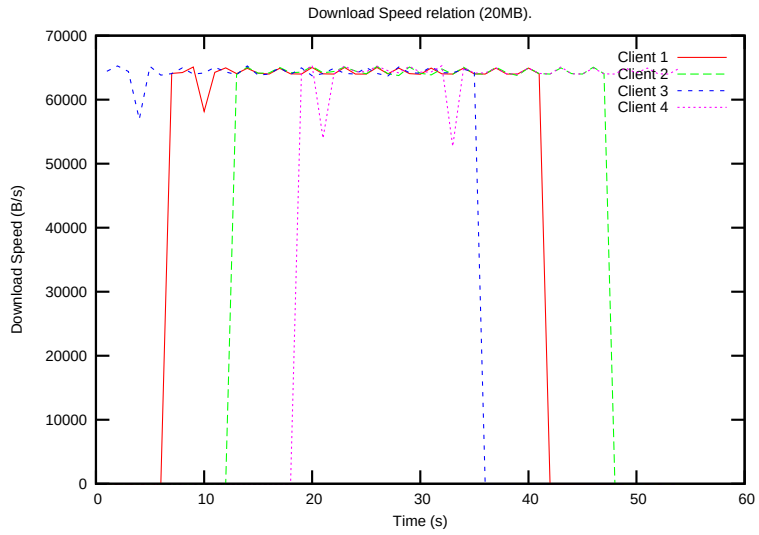

Figura 7.13: Cuatro clientes a 64 KB/s y un ancho de banda de 256 KB/s.

Podemos concluir que en condiciones equitativas para todos los clientes, el sistema se comporta de la forma esperada, repartiendo los recursos de una forma equitativa entre cada uno de los clientes que están en el sistema, independientemente del orden de entrada al mismo.

Tres clientes a 16 KB/s y la entrada de un cuarto a 1024 KB/s con un límite total de 32 KB/s. En este ejemplo, se puede observar el comportamiento equitativo de los cuatro clientes, aunque uno de ellos tenga un ancho de banda mayor. La entrada de un cuarto cliente con un ancho de banda muy superior cuando el ancho de banda es muy limitado. Lo importante a analizar en este caso es si los clientes con un ancho de banda menor pierden su conexión por saturación del canal al aparecer un cliente con gran capacidad de descarga.

Como se observa en la Figura 7.14, el cuarto cliente, que es el que tiene un ancho de banda muy superior, se comporta como un cliente normal, con un ancho de banda similar al otorgado al resto de los clientes. Una vez los demás clientes van finalizando sus descargas, el canal está cada vez menos saturado y el cuarto cliente puede ir aprovechando el ancho de banda libre.

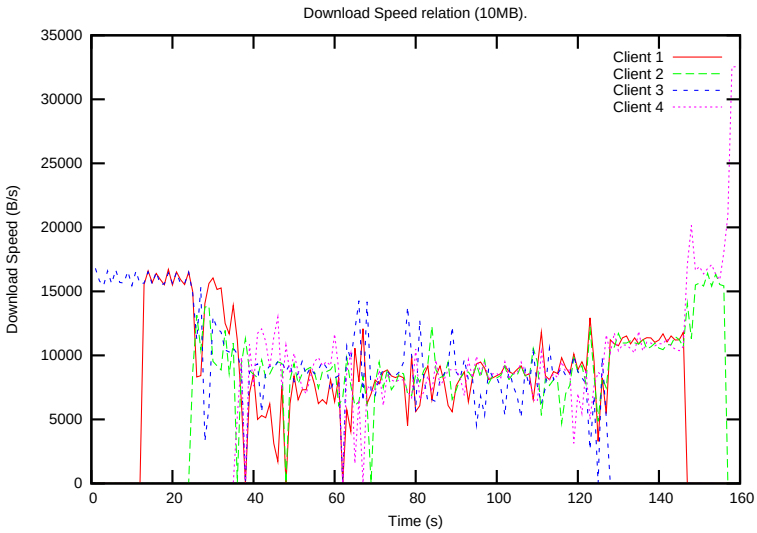

Figura 7.14: Tres clientes a 16 KB/s, un cuarto a 1024 KB/s y un límite de 32 KB/s.

Tres clientes a 16 KB/s y la entrada de un cuarto con 1024 KB/s con un límite total de 128 KB/s. En esta gráfica se quiere observar el comportamiento del sistema una vez entra el cuarto cliente en un sistema no tan limitado como el caso anterior. Ahora subimos el ancho de banda máximo del servidor para que de cabida a los tres primeros clientes, siendo el cuarto, el que podría interferir en los demás al solicitar una gran cantidad de ancho de banda.

En la Figura 7.15, se puede observar como en este caso, el cuarto cliente no consigue llegar a ocupar los 128 KB/s que podría solicitar, dejando a los otros tres clientes que ya estaban conectados, seguir con su descarga de forma normal.

Un cliente a 16 KB/s y las sucesivas entradas de clientes a 128 KB/s con un límite total de 32 KB/s. Aunque ya se ha probado el comportamiento del sistema con diversos clientes con baja velocidad de conexión, y el efecto que tiene la entrada de un cliente con una conexión mucho mayor; no se ha comprobado el caso en que solamente uno de los clientes tenga un ancho de banda reducido. En la Figura 7.16, se comprueba primeramente, como se comporta la existencia de un cliente con poca velocidad de conexión en un entorno en que todos los demás clientes tienen una velocidad superior. En este caso, el ancho de banda total posible es solamente de 32 KB/s.

Como se observa, el ancho de banda se sigue repartiendo de forma equitativa entre todos ellos, siempre sin superar el máximo permitido. No se hacen distinciones entre los cuatro clientes cuando el sistema esta saturado.

Un cliente a 16 KB/s y las sucesivas entradas de clientes a 128 KB/s con un límite total de 512 KB/s. En la Figura 7.17, se puede observar el comportamiento del sistema al entrar los diversos clientes. Como ya se ha

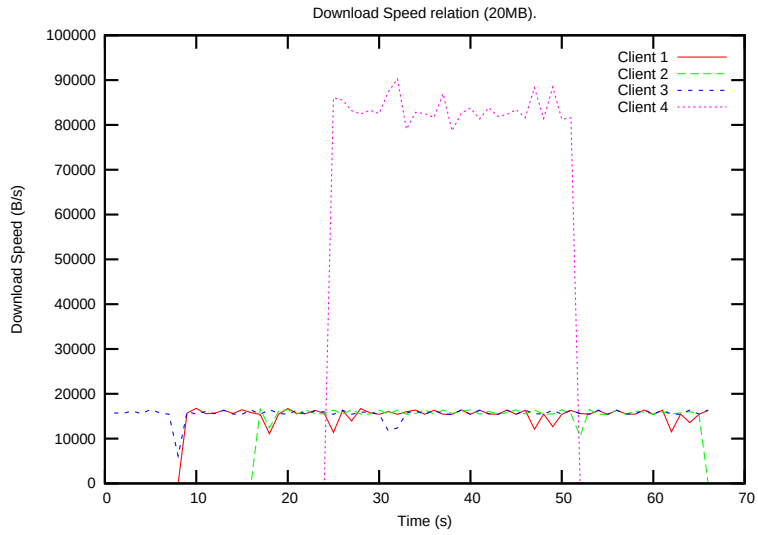

Figura 7.15: Tres clientes con 16 KB/s, un cuarto con 1024 KB/s y un límite de 128 KB/s.

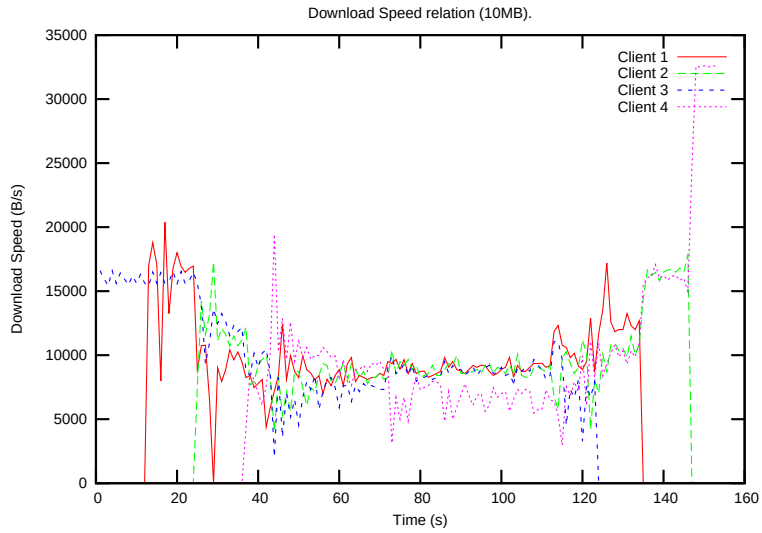

Figura 7.16: Un cliente a 16 KB/s, tres más a 128 KB/s y el límite a 32 KB/s.

comentado anteriormente, se puede observar como con la entrada del último cliente, el ancho de banda lo pierden aquellos clientes con un ancho de banda mayor, mientras que el cliente de 16 KB/s sigue manteniendo su velocidad sin apenas oscilaciones.

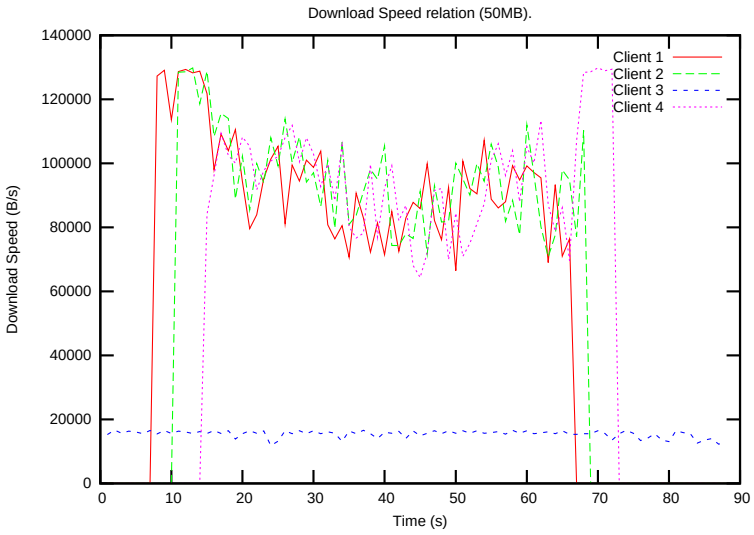

Figura 7.17: Un cliente a 16 KB/s, tres más a 128 KB/s y el límite a 512 KB/s.

7.3.4. Conclusiones obtenidas.

A la vista de los resultados obtenidos y mostrados en las gráficas anteriores, se puede llegar a la conclusión de que, tal como está diseñado el sistema, el comportamiento del mismo es el siguiente:

1. Si el sistema tiene suficiente ancho de banda para dar soporte a todos los clientes conectados, este funciona normalmente.

2. Si el sistema carece del ancho de banda suficiente, la reacción del mismo será:

   a) Si el ancho de banda es muy inferior al necesario, este se reparte equitativamente entre todos los clientes, independientemente del ancho de banda contratado por cada uno.

   b) Si el ancho de banda, aunque inferior al necesario, es suficiente para dar servicio a los clientes con una conexión reducida, los clientes con menor ancho de banda podrán acceder al sistema normalmente. Son los clientes con mayor ancho de banda los que acusan las pérdidas de velocidad.

Por tanto, si se pretende que el sistema tenga una distribución más equitativa y proporcional al ancho de banda de cada cliente, es decir, que se disminuya la velocidad en caso de necesidad a cada uno de los clientes de forma proporcional a su conexión; se deberá añadir programación adicional que controle este caso de manera artificial al comportamiento de la red.

7.4. Compatibilidades con distintos navegadores.

El sistema ha sido probado con tres navegadores, obteniendo resultados totalmente satisfactorios en dos de ellos, mientras que en el tercero hay que realizar algunos ajustes. El primer navegador probado ha sido el Microsoft Internet Explorer, que no ha dado ningún problema en el acceso al sistema y el uso del mismo. El segundo navegador ha sido el Firefox. Este navegador, que ha sido probado en distintos sistemas operativos, no ha tenido tampoco ningún problema y permite el acceso a cualquier parte. Con el tercer navegador, el Konqueror, cada vez que se refresca la ventana del cliente, aparece un mensaje de confirmación del usuario para dar permiso a la recarga de la misma. Esto, que puede ocurrir cada 20 segundos dependiendo de la configuración del sistema, puede llegar a ser muy molesto para el cliente, que si deniega la recarga, perderá su conexión a Internet en unos minutos. Por ello hay que configurar al Konqueror para que no realice estas confirmaciones. Por el resto, funciona correctamente.

Capítulo 8

Futura utilización del portal. Ejemplos.

A continuación se presentan unos ejemplos que muestran algunas aplicaciones prácticas del portal. En concreto, se presentan extensamente dos aplicaciones prácticas del mismo. Los dos sistemas desarrollados se llaman Dulenduè y RuralNet respectivamente. Dulenduè es una aplicación que pretende dar distintos servicios a un turista que llega a una ciudad que desconoce por completo. RuralNet es un portal que permite la conexión a Internet de zonas de difícil cableado mediante la tecnología *Wireless*.

8.1. Dulenduè.

8.1.1. Introducción.

Dulenduè[1] nació como un portal turístico. La idea de este portal, es permitir la conexión gratuita a un sistema de red *Wireless* mediante cualquier dispositivo móvil por la ciudad, ya sea un portátil, una *PDA* o un móvil con tecnología *Wireless*. El sistema, a su vez, calcula la posición del cliente a partir del nodo de acceso al que se conecta. Esto es lo que se llama una aplicación *context-aware*. Por *context-aware* se entiende a la noción de algunas aplicaciones informáticas que se basan en el entorno del dispositivo cliente para proveer servicios relativos a la persona, lugar o tiempo relacionados con el mismo. Es decir un dispositivo que se relaciona con su contexto social o físico[2].

A partir de esta información, el cliente puede acceder a una búsqueda de servicios como si se tratara de un servicio de Páginas Amarillas pero teniendo en cuenta la posición que tiene el cliente en ese momento. Es decir, aparte de seleccionar lo que se desea buscar, el cliente selecciona la distancia máxima aproximada en donde desea encontrar el servicio.

Al cliente, aparte de este servicio gratuito, es interesante ofrecerle una conexión a Internet mediante esta infraestructura, ya sea de pago o no para que realice búsquedas externas o simplemente se comunique mientras pasea por la ciudad.

[1] Significa "¿Dónde estás?" en un dialecto de la Lombardía, en la región de Milán.
[2] Para más información, buscar en la bibliografía el documento [13].

En el caso de los servidores permitidos de forma gratuita, se puede dar acceso a aquellos servidores que alojen las páginas web de aquellos servicios dados de alta. Por ejemplo, la página de un restaurante dado de alta puede ser visible por todos los clientes, tanto si tienen acceso a Internet como si no es así.

8.1.2. Adaptación del sistema.

Aunque el sistema Dulenduè parte de la arquitectura TocToc, son necesarios algunos pasos de adaptación en el diseño para ofrecer las nuevas opciones que incorpora. Esta adaptación consiste en la presencia de nuevos archivos con funcionalidad extra, cambio de imagen del portal o la modificación de algunos ya existentes:

- Nueva funcionalidad:
  - Se generan las páginas *busquedaset.php busqueda.php* y *busquedaresult.php* para dar la nueva funcionalidad al sistema.
  - Se añade el archivo *posicion.inc*: Este archivo contiene toda la información relativa a la posición del cliente. Aquí están las variables que indican la posición al cliente.
- Cambios de imagen:
  - Se modifica el archivo *index.php* para que albergue una presentación. Esta presentación en Flash, obtiene las coordenadas del archivo *posicion.inc* para modificarse de acuerdo con la posición del cliente.
  - Cambiar la configuración de los archivo *colores.inc*, *dulendue.css* y *dulendueXSL.xsl* modificando los colores y estilos para personalizar el portal. Se cambia la imagen de fondo del portal y el *favicon.ico* (*favourite icon*).
- Adaptaciones para mantener la cohesión de la página:
  - Se cambian las funciones que controlan las sesiones de PHP tales como *guarda_sesion* y *lee_sesion* para que alberguen toda la nueva información que tiene un cliente.
  - Se modifican las páginas *registrarse.php* y *perfil.php* para que el usuario pueda introducir en el formulario los nuevos datos. Se modifican las páginas *crear_usuario.php* y *actualiza_usuario.php* para que estos datos se guarden en la base de datos.
  - Se añaden los nuevos cambios a la base de datos, para que alberguen la nueva información.
  - La página *TocToc.php* renombrada a *Dulendue.php* tiene añadido en el menú la opción "búsqueda" que le envía a la página correspondiente.

Con estos cambios se consigue que el portal cambie de la forma esperada. Estos cambios se pueden observar en la sección siguiente.

8.1.3. Ejemplo de uso.

Como el sistema Dulenduè está basado en el sistema TocToc, tiene un funcionamiento muy similar y aparecen las mismas opciones que este.

El inicio de Dulenduè. Si se observa la Figura 8.1, se puede apreciar los cambios que se han introducido en esta nueva versión del sistema TocToc. Primero, lo que se puede encontrar es que se ha añadido un nuevo idioma. Como se ha indicado anteriormente, esta tarea está muy facilitada por la estructura del sistema TocToc.

La presentación, ahora es un *flash* que representa un mapa de la ciudad de Valencia. Una animación marca la zona en la que el sistema deduce que está el cliente.

El resto del sistema, sigue la estructura original de TocToc, pero podría haber sido modificada según los gustos del diseñador.

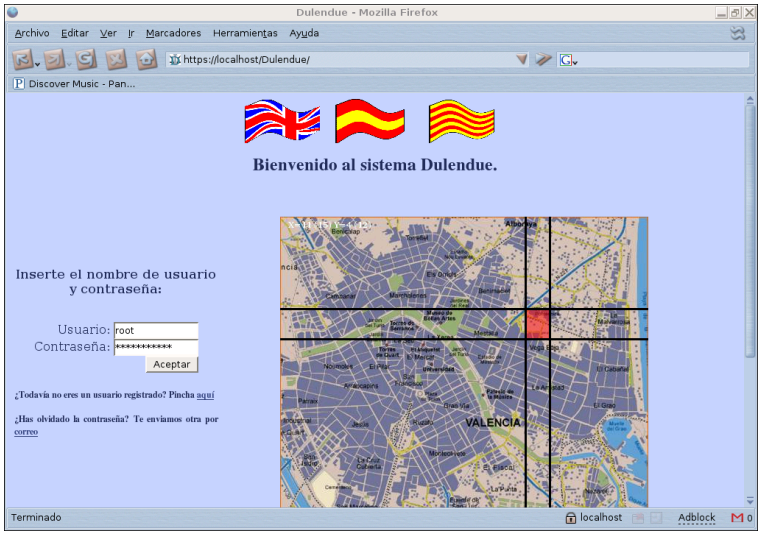

Figura 8.1: La página de inicio de Dulenduè.

Una vez dentro de Dulenduè. Si se conecta un cliente y este da de alta un nombre de usuario, podrá observar la página principal de Dulenduè. Esta sigue manteniendo el formato de TocToc, salvo algunos retoques de imagen y una nueva opción que aparece en el menú: "búsqueda". La opción de búsqueda se explica en la siguiente sección.

Alguna otra novedad que se muestra, es que aparece en el menú superior las coordenadas estimadas del cliente. Estas coordenadas, que pueden resultar poco significativas para el cliente, permite observar si el sistema detecta el cambio de posición para un cliente que esté en movimiento, como por ejemplo, dentro de un coche. O pueden ser cambiadas por el nombre del barrio donde se supone que se encuentra.

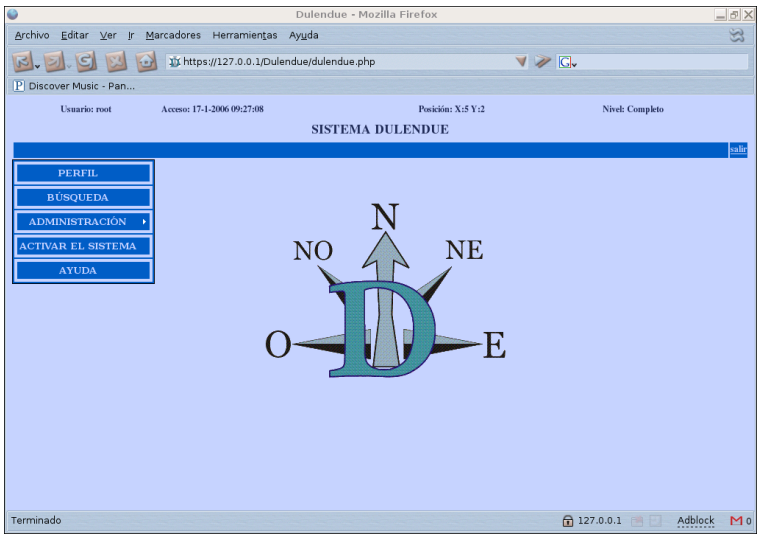

Figura 8.2: Las novedades de Dulenduè.

Novedades en los clientes del sistema Dulenduê. Cuando nos damos de alta en el sistema Dulenduê, se puede observar que aparece un conjunto de opciones llamado "preferencias". En la Figura 8.3 se puede observar que aparecen para seleccionar "Aparcamientos", "Cines", "Museos" y "Restaurantes".

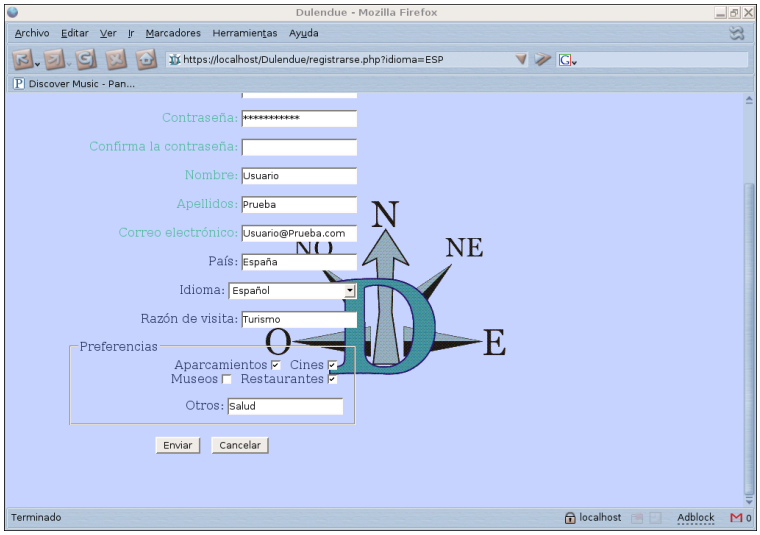

Figura 8.3: Novedades en el registro de clientes.

Estas opciones son generadas de forma dinámica en el sistema y depende del tipo de servicios que estén dados de alta. Aquí un cliente puede elegir lo que quiere que le aparezca en el menú de búsqueda. Por ejemplo, si un cliente

solamente busca algún lugar donde poder comer y suponiendo que el sistema tenga ya un gran número de servicios dados de alta, el cliente aquí puede dejar solamente marcadas las opciones de "Restaurantes" y "Aparcamientos". De esta forma, estos dos servicios serán los únicos que el cliente podrá seleccionar en el menú de búsqueda. Si más tarde aparece un servicio nuevo, este aparecerá por defecto seleccionado, por lo que el cliente explícitamente tendrá que indicar que no quiere saber nada de él.

El campo "Otros" permite al sistema saber aquellas preferencias del cliente que todavía no están contempladas en el sistema. En este ejemplo, al cliente le resultaría imposible encontrar una farmacia o un hospital. Esta indicación puede ayudar al diseñador a añadir aquellos servicios que tengan demanda y no hayan sido incluidos.

Realizar una búsqueda de un servicio. Si entramos en la opción de búsqueda, nos sale un conjunto de preguntas para acotar los resultados a mostrar. Hay que señalar que a diferencia de otros motores de búsqueda, aquí no se puede seleccionar un servicio en concreto, sino que se pretende seleccionar cualquier servicio dentro de un radio y que cumple unas características adecuadas. Esto se aprecia mejor si se observa la Figura 8.4. Como se puede observar en la Figura, en el menú de la izquierda, se puede seleccionar los distintos servicios disponibles (en el ejemplo, se han seleccionado "Restaurantes"). Se puede seleccionar también una aproximación del precio máximo que se quiere invertir en el servicio (en el ejemplo menos de 40 euros) y la distancia máxima a la que se quiere que se encuentre el servicio. Que en este caso son 200 metros.

A la derecha se muestra el conjunto de resultados obtenidos, separados por páginas. Cada resultado tiene un conjunto de información relevante, como la dirección exacta y la página web. A la página web, en principio solamente se podrá acceder si el cliente tiene conexión a Internet o si el dueño del servicio paga un extra y su servidor sea uno de los de acceso gratuito.

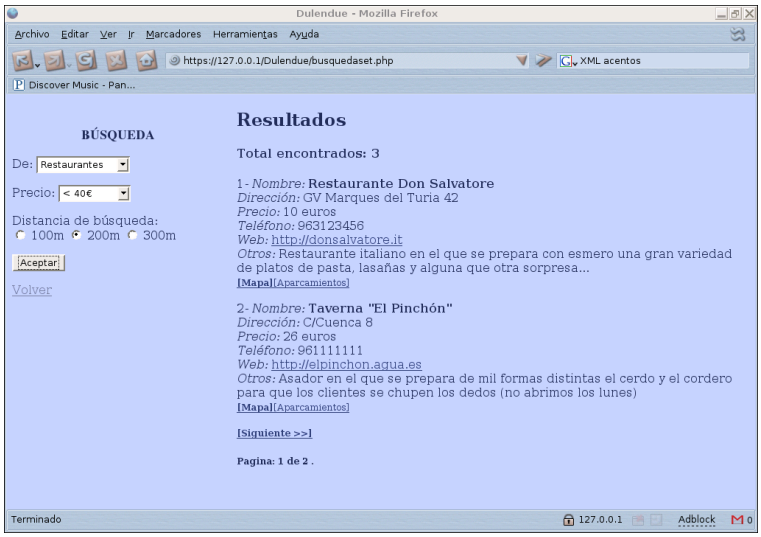

Figura 8.4: Realizando una búsqueda.

Debajo de cada servicio, aparecen dos enlaces que permiten realizar una segunda búsqueda. La opción de "aparcamiento" ordena al sistema buscar un aparcamiento público por la zona en la que reside el servicio seleccionado. La opción de "mapa" realiza una consulta a un servidor externo, en este caso la Vía Michelín. Indicando la localización exacta del servicio en un plano.

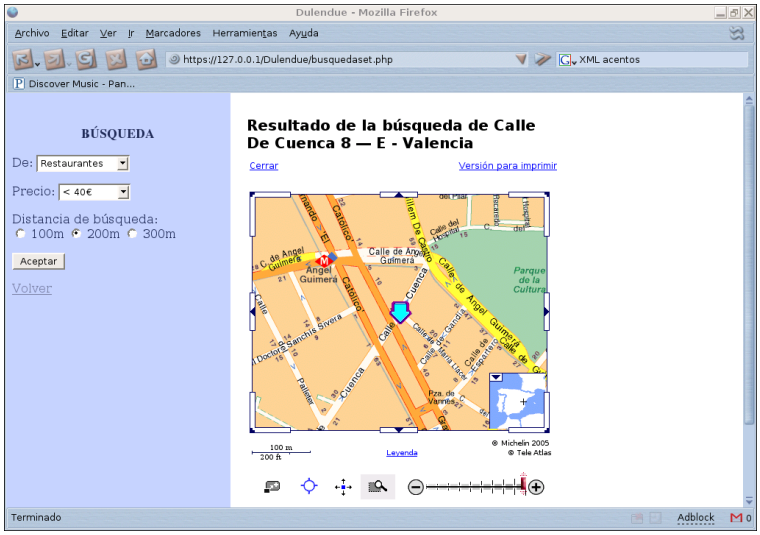

Figura 8.5: Mapa de la búsqueda.

Para que esto funcione, se tiene que habilitar el servidor de la Vía Michelín y los distintos servidores que muestran los mapas en el sistema para que los clientes sin acceso a Internet también puedan acceder a ellos. Este no es un servicio gratuito, por lo que se tiene que contratar a la Vía Michelín o cualquier otra empresa similar. Una vez contratado, Vía Michelín permite posibilidades de mejora e integración con el portal, tales como el uso de una hoja de estilos en cascada (CSS) propia para mantener la imagen y color del mismo.

Un ejemplo de página de ayuda. En todos los sistemas conviene completar la ayuda disponible que trae TocToc para explicar un poco el funcionamiento que pueda tener la páginas. Por ejemplo, en Dulenduè, quizá lo más complicado de utilizar sea el menú de búsqueda. Añadir la información adecuada en el sistema puede facilitar al usuario la utilización del mismo.

El sistema Dulenduè aprovecha la estructura de ayuda que ofrece TocToc, añadiendo el contenido específico de este sistema.

8.1.4. El problema de la localización.

Aunque el sistema esta diseñado, falta por solventar cómo este deduce la posición del cliente.[3]

La idea para obtener esto, consiste en tratar las propiedades de la señal recibida del cliente, tales como la fuerza de la misma o su dirección aproximada.

[3] De esta parte, se encarga Alejandro Bellido Server en su proyecto final de carrera.

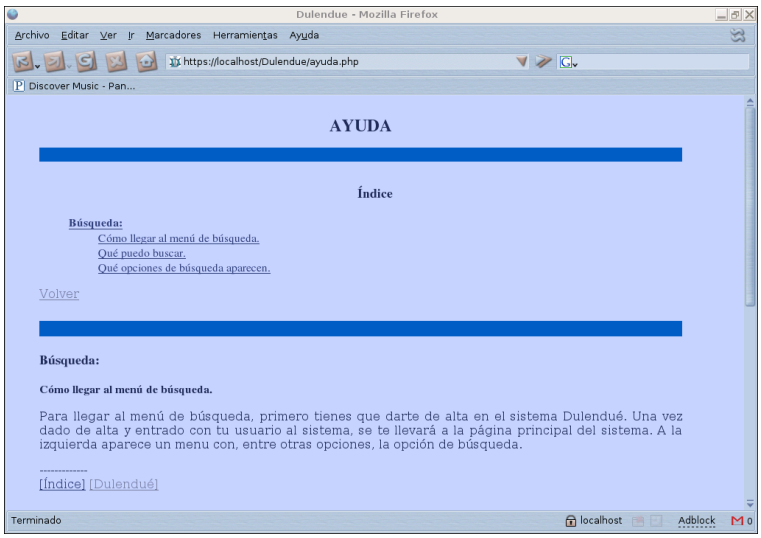

Figura 8.6: La ayuda de Dulenduè.

Por ejemplo, en la Figura 8.7, se representa el método de localización basado en dos antenas direccionales en las que se puede obtener un ángulo aproximado de donde llega la señal. Por tanto, mediante un ejercicio sencillo de triangulación, se puede obtener la posición.

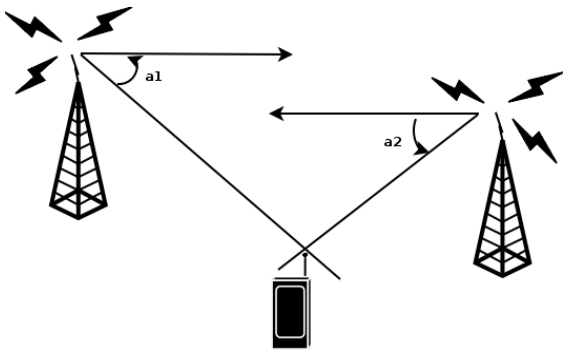

Figura 8.7: Ejemplo de localización de un cliente.

En la Figura 8.8, sin embargo, se utiliza la fuerza de la señal que llega a tres nodos y con ella se busca la localización posible en cada uno de ellos. El punto de intersección dará la posición del cliente.

La elección entre una metodología u otra viene determinada por la tecnología que se emplee a la hora de realizar la infraestructura necesaria. Para más información se puede consultar [9].

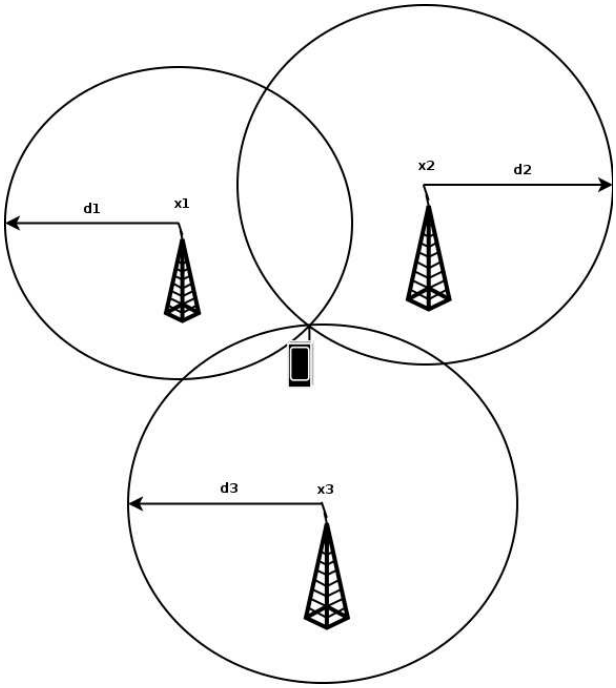

Figura 8.8: Otro ejemplo de localización.

8.2. RuralNet.

RuralNet es un portal que pretende dar conexión a las zonas rurales mas alejadas de los centros urbanos. Generalmente estas zonas tienen difícil la posibilidad de un cableado para recibir conexiones a Internet de alta velocidad. Esta infraestructura suele ser muy costosas de instalar y resulta poco rentable para zonas poco pobladas. Una alternativa a esto, sería la instalación de una infraestructura inalámbrica que resuelva el problema de la distancia, dando acceso a cada habitante de la zona que disponga una tarjeta adecuada en su equipo. De esta forma, el cliente puede tener acceso a distintos servicios impensables de otro modo, como Internet de banda ancha, vídeo digital o incluso teléfono mediante VoIP si no se dispone una forma más tradicional. En la Figura 8.9 se observa un esquema explicativo.

TocToc puede servir a este propósito, ofreciendo la interfaz de control de usuarios para que estos se den de alta en el sistema y puedan comprar una conexión a Internet. Con este propósito nace RuralNet. Este portal es una representación del sistema TocToc para el ámbito rural. Puede ser configurado para ofrecer páginas gratuitas locales, como predicción del tiempo, estado del tráfico, servicio de noticias o consultas agrarias. Si el cliente lo desea, también puede ofrecer algunos servicios más elevados como comercio electrónico (tales como subastas electrónicas, oferta de estancias turísticas,...), telemedicina, incluyendo en este término las consultas a distancia para pacientes de difícil acceso y de escasa movilidad; o el apoyo a médicos menos cualificados, realizado por otros con el conocimiento necesario pero que trabajan a una gran distancia.

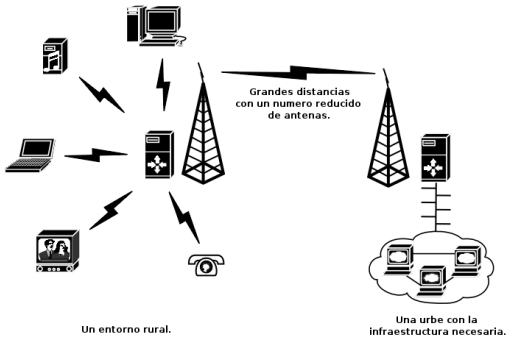

Figura 8.9: La idea de una red rural.

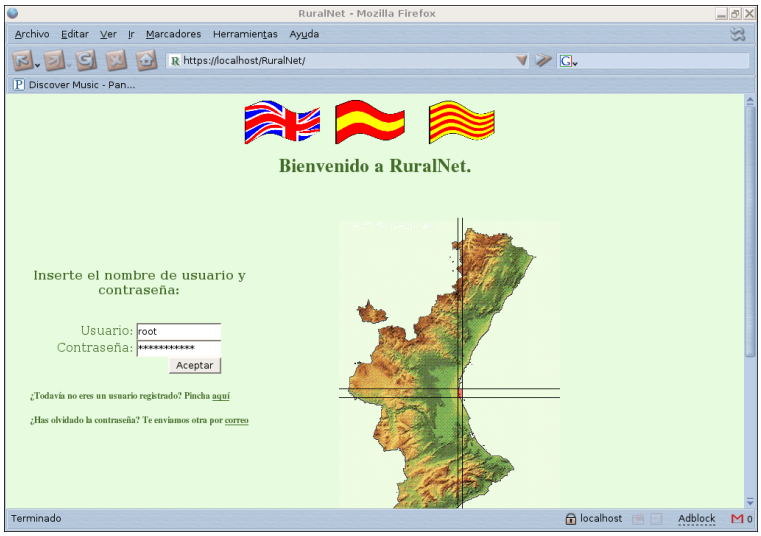

Figura 8.10: Presentación de RuralNet.

El acceso a Internet sería el que ya tiene diseñado el sistema TocToc y el resto de servicios, tales como VoIP, serían programarlos a través del sistema de servidores permitidos que ofrece TocToc. Las Figuras 8.10 y 8.11 muestran un ejemplo de lo que podría ser RuralNet.

8.3. Incompatibilidades para superponer varios de estos ejemplos.

Un último detalle sería comentar la posibilidad de implementar en una misma situación varios de estos ejemplos y albergarlos en una misma máquina. La respuesta a esta pregunta sería decir que como máximo puede existir uno de estos sistemas por servidor, o como mucho uno por tarjeta *wireless* o punto de acceso. Es necesario que el servidor DHCP distinga los clientes y le sirva

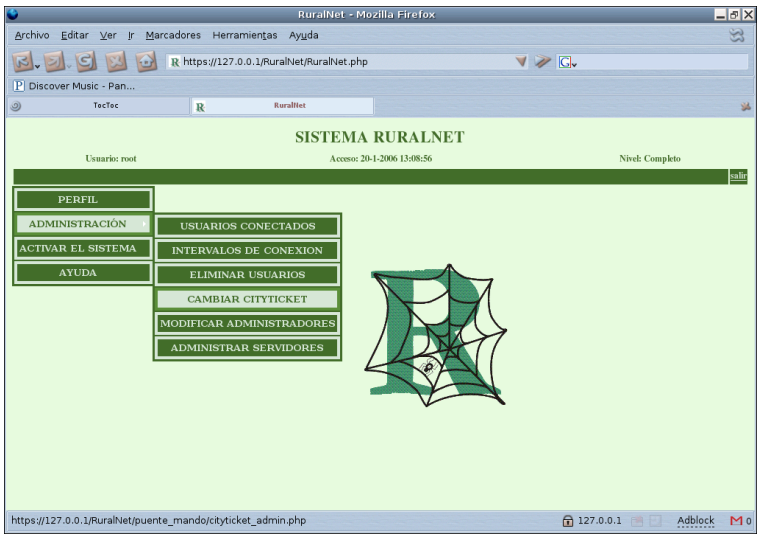

Figura 8.11: Dentro de RuralNet.

una IP dependiendo del sistema al que se quiera acceder. Es decir, debería de poder crearse distintas redes completamente independientes. Así los *scripts* de ejecución del sistema diferenciarán a cual de ellos corresponde tratar cada cliente.

Lo que no es posible hacer es que un mismo cliente puede conectarse a la vez a dos portales que estén en una misma máquina, ya que la sesión de PHP se confunde y mezcla información.

Capítulo 9

Trabajo futuro.

Aunque la arquitectura TocToc ha sido ya comentada y estudiada en los capítulos anteriores, existen siempre ampliaciones que se pueden plantear para una futura revisión. Se puede mejorar la gestión de la velocidad de los clientes haciéndola más precisa si fuera necesario (pero con un coste computacional mayor) o se puede exportar el sistema a otras tecnologías. También se puede filtrar no solo los clientes, sino el tráfico de cada uno de ellos pudiendo dar distintas prioridades a las distintas aplicaciones si la red está muy saturada.

Existen también, algunos problemas derivados de la conexión *wireless* que son comunes en todos estos sistemas y que pueden ocasionar algunas dificultades a la hora de exportar esta arquitectura. Estos problemas se comentan para que se tengan en cuenta a la hora de instalar el sistema e intentar evitarlos en la medida de lo posible.

9.1. Utilización de IMQ.

El IMQ o *Intermediate Queueing Device* es una utilidad que se combina junto al TC para dar a este una mayor precisión. El sistema de colas que se ha utilizado en la arquitectura ha sido el llamado SFQ, que resulta un tanto impreciso para controlar el acceso de la red a los distintos usuarios, a cambio de un mejor rendimiento. Esto es debido a que SFQ, para dar la velocidad correspondiente a cada cliente, realiza aproximaciones estadísticas y regula las peticiones al ISP (*Internet Service Provider)* de acuerdo con esas aproximaciones, pero no controla realmente el tráfico de entrada. Esto se muestra en la Figura 9.1.

Además, existe la limitación de que estas colas solamente se pueden asignar a un dispositivo red y no globalmente al sistema.

IMQ sin embargo, crea un dispositivo virtual en al servidor convirtiéndole en su propio ISP, de forma que solamente pide a la red externa los paquetes realmente necesarios. De esta forma se tiene pleno control sobre el ISP pudiendo controlar exáctamente el tráfico.

Como se puede observar en la Figura 9.2, realmente se piden las peticiones correspondientes, por lo que se regula el tráfico de entrada correctamente, tarea muy limitada con otro tipo de colas.

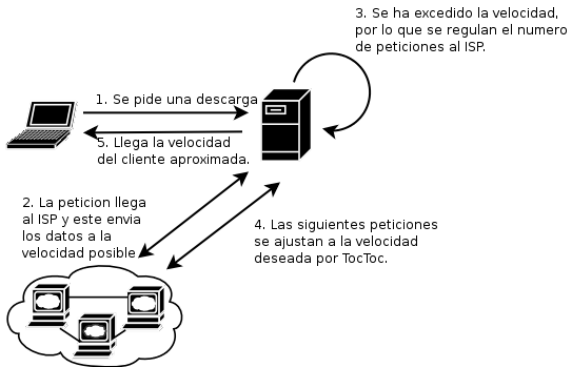

Figura 9.1: Funcionamiento estándar de QDISC.

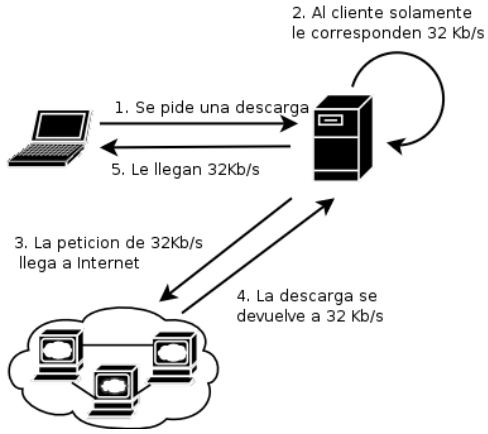

Figura 9.2: Funcionamiento de IMQ

9.2. Reparto equilibrado del ancho de banda.

Como se ha observado en el capítulo de experimentación, en entornos con ancho de banda limitado se ha observado que el sistema tiende a repartir el ancho de banda de forma equitativa entre todos los clientes. Esto que permite un acceso a cada uno de los clientes, desencadena en que, en el caso de que el sistema esté muy saturado, los clientes que contraten un ancho de banda superior obtengan a la larga el mismo ancho de banda que el resto de clientes que realizan un desembolso económico menor. Esto se muestra en la Figura 9.3.

Esto se puede solventar si, a la hora de dar conexión a un cliente, se regula la conexión mediante una implementación extra de manera que, en vez de solicitar su ancho de banda completo, el cliente solicita un ancho de banda proporcional al total disponible en el sistema respecto al conjunto de clientes que ya está dado de alta en el sistema. Es decir, asegurarse que aquel que tenga contratado el doble de ancho de banda, obtenga proporcionalmente el doble de ancho de banda, limitando el resto de los clientes para evitar la saturación del sistema. De esta forma se consigue que los clientes reciben, si no lo que les corresponde

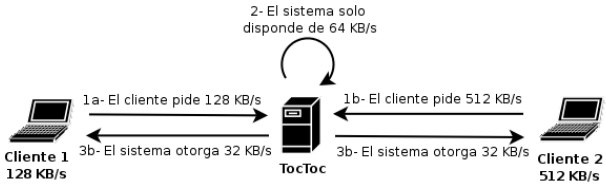

Figura 9.3: Reparto equitativo del ancho de banda.

por imposibilidad del sistema, una parte proporcional a lo que están pagando. Así se justifica el desembolso extra del cliente.

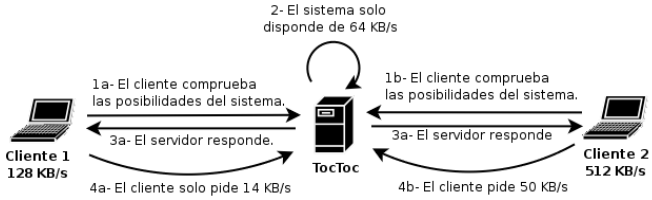

Figura 9.4: Reparto proporcional del ancho de banda.

En la Figura 9.4, como se observa, el cliente 2 con un ancho de banda superior, aunque no puede obtener todo el que ha contratado, tiene una mayor capacidad que el cliente 1.

9.3. Regulación del acceso a Internet de los programas del cliente.

El programa TC de Linux permite, aparte de todo lo descrito anteriormente, realizar una comprobación de la cabecera de las tramas que pasan a través de él. De esta forma, se pueden modificar las prioridades de ciertos paquetes para evitar que saturen a los demás clientes e impidan su acceso. Por ejemplo, si se le da una prioridad mínima a aquellos paquetes que pertenezcan a un programa P2P, permitimos que un cliente con poco ancho de banda acceda a la red, a pesar de que varios clientes con un ancho de banda superior dejen sus equipos encendidos utilizando un programa de la clase P2P (como el Emule).

Otro ejemplo, si el sistema pretende usarse con programas de VoIP, (transmisión del habla por Internet), sería dar a estos paquetes la máxima prioridad posible en el sistema, ya que un retraso en los mismos puede imposibilitar una conversación.

9.4. Otras posibles aplicaciones del sistema.

Una forma de ampliar la arquitectura TocToc podría ser añadiendo nuevas funcionalidades que le permitan abarcar un campo más amplio o el paso a otras tecnologías distintas a las usadas hasta ahora. Esto último podría permitir

la instalación de esta arquitectura en otros entornos. Un ejemplo se lista a continuación.

9.4.1. Museo de la Ciudad de las Artes y las Ciencias. Paso a *Bluetooth.*

No es necesario que todo este sistema abarque una extensa zona geográfica como una ciudad o una comunidad entera. El sistema puede ser igual de efectivo si se cambia el soporte físico. Un ejemplo de esto, sería el paso del sistema de un soporte *wireless* a un soporte *Bluetooth*, teniendo las mismas características que antes pero en un entorno más reducido (recordemos que el sistema *Bluetooth* soporta TCP/IP).

De esta forma se puede adaptar el sistema Dulenduè a una nave industrial o una habitación. Por ejemplo, en el museo Príncipe Felipe de la Ciudad de las Artes y Ciencias de Valencia, se puede elaborar un sistema, de forma que los clientes, con la entrada, reciben una *PDA* a modo de préstamo con acceso *Bluetooth*. De esta forma, el sistema sabrá en todo momento donde está situado el cliente en todo momento con un error máximo de unos pocos metros, ya que el protocolo usado en *Bluetooth* le da un alcance muy limitado a estos aparatos y es fácil deducir a que nodo está más próximo. Así se le puede ir presentando información de la atracción a la que se aproxime de una forma automática, con una explicación detallada que puede leer el poseedor de la *PDA* sin agobios ni colas. Por supuesto, el acceso a Internet, puede ser utilizado limitándolo a ciertos servidores de forma que se puede complementar la información mediante artículos científicos, portales de ciencia o lo que el museo considere oportuno. No existirá la posibilidad de que el cliente use la *PDA* para otros fines, ya que estará el sistema totalmente limitado a las posibilidades que se le indiquen explícitamente.

9.5. Problemas con la conexión *wireless.*

9.5.1. Desconexiones de red y pérdidas de descargas.

El principal problema consiste en que un cliente puede estar desplazándose por la zona, puede estar en una zona que la cobertura no sea óptima o puede perder señal por cualquier otro motivo. La forma con la que esta ideado el portal, incluye un sistema que libera al cliente si el servidor no tiene señal del mismo durante un tiempo, echándolo del mismo. Esto implica que una perdida continuada de señal será reconocida por el sistema como una desconexión del cliente y por tanto este cuando recupere la señal, tendrá que volver a introducir su usuario en el mismo.

La principal molestia viene dado por aquellos clientes que realicen descargas continuas de la red, ya que estos tendrán que comenzar de nuevo la descarga. Si este sistema se esta ofreciendo como ISP (*Internet Service Provider*) puede ser un motivo más que suficiente para perder un numero importante de clientes.

9.5.2. Problemas de distancia.

Un problema que tienen las redes basadas en ondas de radio como la 802.11, según Ron Olexa [9], es conocido como el problema de cercanía/lejanía. En

esta situación, las señales más fuertes enmascaran las más débiles, llevando a un consumo desproporcionado de ancho de banda por aquellos clientes que tienen una señal más fuerte, como por ejemplo, aquellos que están más cerca. Una posible solución consiste en usar unas tarjetas más potentes o la conexión de estas a antenas externas, para aquellos clientes que estén en las zonas más desfavorecidas por la señal. Esto, por contra, supone un desembolso económico extra.

9.5.3. Otras redes en el canal: solapamientos e interferencias.

El problema. El protocolo CSMA/CA produce algunos problemas en áreas en donde múltiple estaciones base comparten un canal común [9]. Puede ocurrir que las estaciones base se escuchen unas a otras pero los clientes, debido a una localización apantallada no puedan escuchar a la estación adecuada y comiencen a transmitir a otras estaciones. Esto provoca interferencias en la transmisión ya que una estación base no puede discriminar entre las señales deseadas y las no deseadas. El resultado es la pérdida de paquetes y la necesidad de retransmisiones.

El por qué del problema. Para entender el problema, primero hay que entender la frecuencia a la que funciona y el ancho del canal. La frecuencia libre que comprende la banda de 2,4 GHz utilizada por los dispositivos *wireless* está subdividida en canales, que varían de acuerdo a las leyes de los diferentes países que los regulan. El estándar IEEE define una separación mínima entre canales de 5 MHz, por lo que, empezando de 2.412 GHz. Esto se puede observar mejor en la tabla 9.1. En dicha tabla (obtenida de [9]), las "X" en el país indica que canal está disponible al público por normativas locales y reservas del ancho de banda para uso del propio estado. También hay que señalar que no todos los dispositivos llegan al uso de los 14 canales por cuestiones de diseño.

CANALES	FRECUENCIA	EEUU	EUROPA
Canal 01	2.412 GHz	X	X
Canal 02	2.437 GHz.	X	X
Canal 03	2.422 GHz.	X	X
Canal 04	2.427 GHz.	X	X
Canal 05	2.432 GHz.	X	X
Canal 06	2.437 GHz.	X	X
Canal 07	2.442 GHz.	X	X
Canal 08	2.447 GHz.	X	X
Canal 09	2.452 GHz.	X	X
Canal 10	2.457 GHz.	X	X
Canal 11	2.462 GHz.	X	X
Canal 12	2.467 GHz.		X
Canal 13	2.472 GHz.		X
Canal 14	2.477 GHz.		X

Cuadro 9.1: Tabla de frecuencia y canales Wireless.

Cada canal necesita un ancho de banda de 22 MHz para transmitir la información, por lo que se produce un inevitable solapamiento de varios canales contiguos. Para evitar interferencias en presencia de varios puntos de acceso cercanos, estos deberían estar en canales no solapables, como por ejemplo: 2, 7 y 12

Como se puede observar, esto realmente solamente deja tres canales totalmente libres de interferencia. Teniendo en cuenta que este sistema puede dar cobertura completa a una ciudad o incluso a una comunidad o país, la probabilidad que se solape con algún canal local es realmente alta. Esto se traduce en perdidas de velocidad, caída de señal y otros malestares para el cliente.

Soluciones. Una forma de evitar esto sería implementar para las redes inalámbricas el protocolo CSMA/CD, que tiene detección de colisiones. Un análisis de la compañía Breezecom [1] indica que esto no resulta viable por:

1. Para implementar este mecanismo sería necesario implementar un sistema de radio *Full Duplex* capaz de transmitir y recibir a la vez. Esto incrementaría el precio considerablemente.

2. En un entorno sin cables, no se puede suponer que cada estación escucha a todas las demás (y esto es una premisa básica para la detección de errores), lo que significa que, aunque el medio este libre para el emisor, no tiene por que estar libre para el receptor.

Con los estándares actuales (802.11a, 802.11b o 802.11g), la solución es prácticamente inexistente. El rango de canales no puede ser aumentado ni modificada la distancia entre ellos. La esperanza reside en como quede en parte el nuevo estándar 802.11n ([3]), que todavía no ha salido a la luz desde que se propuso en enero del 2004 por discusiones técnicas, y que no se espera que aparezca hasta finales del 2006. Este estándar, que trabaja en la banda de 2,4 GHz y/o 5 GHz ofrece entre otras cosas una velocidad 50 veces superior.

La solución que queda por ahora, mientras se espera la llegada del estándar 802.11n, según Ron Olexa [9], pasa por mejorar la planificación, la reducción de la intensidad de señal según sea necesario o la implementación RTS/CTS (*Request to Send / Clear to Send)* en las redes 802.11. Esto consiste, según Jim Geier [5], en que una estación antes de transmitir información, envía un paquete RTS al punto de acceso y no puede comenzar a enviar hasta que no recibe la señal CTS. El paquete CTS alerta a las otras estaciones para que dejen libre el medio y no interfieran.

9.5.4. Acceso a los datos de otros clientes.

Si varios clientes se conectan a una misma red Wireless que no posee encriptación ni WEP ni WAP, implica que los datos que se mueven a través de ella pueden ser interceptados por algún cliente malicioso. Esto se soluciona, en parte, con el protocolo SSL que utiliza el portal TocToc pero que no ocurre en todas las páginas a las que después puede acceder el cliente. Para solventar esto, habría que idear alguna forma de encriptación de datos entre los clientes y el servidor que les da acceso, como realizar una VPN.

Otro problema menor, es que pueden existir varios clientes conectados a la red, que dejarán expuestas sus carpetas compartidas a la red o sus propios equipos si no se adoptan las medidas necesarias en cada cliente. Se puede permitir la distribución de *firewall* gratuitos mediante la red, para que los clientes que no dispongan de uno, protejan sus equipos.

9.5.5. Debilidades ante *Net Flooding.*

La red también queda expuesta a que cualquier equipo que se conecte a la red, incluso sin darse de alta en el portal, puede inundar de tráfico malicioso la misma (*Net Flooding*), colapsándola y dejando a todos los clientes sin ancho de banda suficiente para acceder a la misma. Este es un problema difícil de solucionar ya que las alternativas que quedan son echar esa *IP* de la red, pero nada impide que se vuelva a conectar a la misma ocupando una nueva *IP*. Posiblemente un control mediante *MAC* a un equipo que se detecte que envíe demasiado tráfico innecesario a la red puede dar una solución temporal al problema (recordemos que muchas tarjetas hoy en día permiten enmascarar su *MAC*). Este problema, existe hoy en día en cualquier red y por tanto, en el sistema TocToc.

9.5.6. Algunos problemas más.

¿Que pasa si algún cliente entra con una IP que no está en el rango DHCP? Si el sistema no está preparado, este entrará y el sistema no lo controlará, lo que implica que un cliente que conozca la dirección del *Gateway* puede acceder a Internet si selecciona una IP válida que esté fuera del rango del sistema. Esto debe ser controlado por la máquina servidor o por el punto de acceso y permitir únicamente el acceso a los clientes que entren de la forma adecuada.

Capítulo 10

Conclusiones.

En este documento se ha presentado la arquitectura TocToc. Como ya se ha dicho, TocToc es una arquitectura que pretende dar acceso a Internet mediante la instalación de una red *Wireless*. Permite el control independiente de cada usuario que está dado de alta en el sistema y funciona de manera transparente para ellos.

La necesidad de esta arquitectura viene determinada en los tiempos actuales, por el rápido desarrollo de las telecomunicaciones y la necesidad cada vez mayor de estar conectada la población al torrente de información que es Internet. Existen por contra, zonas desfavorecidas a nivel tecnológico debido a su situación geográfica, ya sea por estar en un lugar apartado o en un terreno agreste, de manera que resulta casi imposible la instalación de la infraestructura necesaria para acercar hasta allí las nuevas tecnologías.

Todo esto se puede evitar si se sustituye el sistema tradicional de cableado por las nuevas tecnologías inalámbricas. Sin embargo, con esta infraestructura surgen nuevos dilemas, tales como el control de los clientes, pues al ser un medio no guiado, cualquiera dentro del campo de acción puede acceder al mismo.

Desde hace unos años, han aparecido distintas soluciones para controlar la autenticación de usuarios y regular de esta forma el acceso a la red. Con este trabajo se ha abordado una solución adicional a este problema. Se ha desarrollado, en primer lugar una arquitectura que permita la implantación de esta idea, generando una estructura de red que da soporte a diversos clientes mediante una conexión *Ad-Hoc*. Con esta base, se ha desarrollado todo un portal que permita el control de usuarios, permitiendo el registro automático de estos en la base de datos y el control de la conexión de cada uno de ellos. Todo esto de una forma fácilmente integrable con otros sistemas, de manera que un administrador pueda controlar todas las opciones del sistema desde una interfaz sencilla y completa.

Los resultados obtenidos en las pruebas realizadas muestran que el sistema es funcional, alcanzando los objetivos de la regulación de las conexiones de los clientes de manera independiente, permitiendo velocidades personalizadas para cada uno de ellos. Además, se permite el acceso de todos los clientes a aquellos servidores externos que sean designados por el administrador para este propósito, pudiendo ofertar por separado un conjunto de servicios gratuitos. Las pruebas realizadas demuestran un comportamiento deseable tanto en condiciones normales, como en una red saturada. No existen clientes que tengan imposibilitado el acceso a la red debido a un exceso de tráfico generado por

otros clientes con mayores privilegios, repartiendo los recursos disponibles entre todos ellos si resulta necesario.

Como se ha mostrado en dos ejemplos propuestos, la adaptación del sistema a las necesidades del cliente resulta sencilla. Además, el hecho de que este sistema esté basado completamente en software libre, y que las herramientas que utiliza también lo sean, permiten además su implantación de una forma económicamente viable.

Bibliografía

[1] Breeze Wireless Communication Ltd, http://www.breezecom.com. *IEEE 802.11 Technical Tutorial.*

[2] Chillispot. http://www.chillispot.org/.

[3] ComunicacionesWorld. La industria se une para desbloquear el desarrollo de 802.11n. http://www.canariaswireless.net, October 2005.

[4] Departamento de Teoria de la Señal y Comunicaciones perteneciente a la Universidad Rey Juan Carlos. Centro experimental de comunicaciones inalámbricas. "http://www.tsc.urjc.es/Investigacion/CECI.htm".

[5] John W. Eaton. Gnu octave.

[6] Jim Geier. Improving wlan performance with rts/cts. http://www.wifiplanet.com/tutorials/article.php/1445641, 13 2002.

[7] Bert Hubert. *Linux Advanced Routing & Traffic Control HOWTO.* http://lartc.org/, 1.43 edition, 10 2003.

[8] Pello Xabier Altadill Izura. *IPtables, manual práctico.* http://www.pello.info/filez/firewall/iptables.html, 1.2 edition, 8 2003.

[9] Nocat. http://nocat.net.

[10] Ron Olexa. *Implementing 802.11, 802.16 and 802.20 Wireless Network.* Elsevier, 2005.

[11] PatronSoft. Firstspot. http://www.patronsoft.com/firstspot/.

[12] James B. Rawlings. Octave. http://www.octave.org/.

[13] Tushar Sachdev. Envisioning a portal solution, August 2005.

[14] SearchMobileComputing. Captive portal. http://www.searchMobileComputing.com.

[15] Paul Dourish Thomas P. Moran. Introduccion to this special issue on context aware computing. http://hci-journal.com/editorial/si-context-aware-intro.pdf.

[16] Wifidog. http://dev.wifidog.org/.

[17] Wikipedia. Captive portal. http://en.wikipedia.org/wiki/Captive_portal.

Apéndice A

Instalación y configuración de TocToc.

Este capítulo tiene como finalidad instruir al lector en la instalación y configuración del sistema TocToc. Así como presentar los requisitos necesarios en el sistema para el correcto funcionamiento del mismo.

A.1. Requisitos.

Para el correcto funcionamiento del sistema TocToc, es necesario tener ya instalados los siguientes programas:

1. Un sistema de cifrado seguro como OpenSSL para dar seguridad a los clientes.
2. Servidor Apache 2.0 o superior. Dicho servidor deberá tener activado la conexión segura con SSL. Cómo se ha configurado la instalación del Apache para este proyecto se muestra de una forma detallada en el apéndice de este documento.
3. MySQL 4.1.10 o superior.
4. PHP 5.0 o superior. Este debe ser configurado para el uso del servidor Apache y MySQL. En el apéndice existe un apartado por si no estas familiarizado con la instalación de PHP.
5. Servidor DHCP propio.
6. El paquete *iproute*. Este paquete aparece en las distribuciones de Linux, por lo que consulte la distribución utilizada para saber como instalarlo.

A.2. Configuración del servidor Apache.

Aparte de la configuración normal del servidor Apache, hay que añadir unas modificaciones no convencionales para que el sistema TocToc funcione correctamente. Como configuración normal se entiende la generación de una carpeta

ligada al mismo que contenga todas las páginas web del servidor, las opciones necesarias para que interprete los documentos PHP y la información relativa al administrador[1].

Como primera modificación extra, hay que descomentar la linea que aparece en httpd.conf: *"ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var"* y modificarla por la siguiente: *"ErrorDocument 404" /error/error.php.* Después de esto añadir los documentos *"error.php" y "error_original.php",* que se incluye junto los archivos del sistema TocToc, en la carpeta *"/Directorio_apache/-errors/"*. Esta página, simplemente redirecciona al cliente a la página de login del sistema. De esta forma el sistema se asegura que cuando capture un cliente que no este dado de alta, siempre se le muestre la página correspondiente y no salga un error. Conviene revisar esta corta página para observar que el enlace de redirección apunta a donde debe.

Una vez acabado esto, hay que realizar exactamente lo mismo para el error 403, de forma que impidamos el listado del contenido del directorio web y lo enviemos a la hoja de registro. Para desactivar la opción por defecto de listar contenidos y que aparezca el error 403 se debe comentar la línea *Options Indexes FollowSymLinks* que existe dentro de la declaración del directorio web. Hay que tener cuidado con estas opciones si se desea albergar múltiples páginas web en este servidor ya que los mensajes de error serán modificadas para todas.

Otro punto a tener en cuenta es que, debido a que el servidor Apache tiene que utilizar algunos comandos restringidos al root, se debe dar permisos al mismo mediante el fichero *"/etc/sudoers"* añadiendo la siguiente línea: *"apache ALL=NOPASSWD:/sbin/iptables, /sbin/tc, /bin/sh"* que le permite ejecutar los comandos TC, IPTABLES y SH[2] necesarios para el sistema.

Por último, recordar que el servidor Apache tiene un usuario en el sistema UNIX asociado, normalmente *apache* o *nobody*. Los scripts del sistema TocToc están preparados para que el usuario del servidor sea el denominado *apache*. Por ello, hay que modificar la línea *user* del archivo *httpd.conf* y colocar el usuario *apache*. Posiblemente haya que crear ese usuario en el sistema y dar, además, permisos de ejecución a los scripts del sistema para este usuario. Generalmente, el no hacer esto, no muestra ningún error al ejecutar el sistema, simplemente este se limita a ignorar los scripts para los cuales no tiene permisos, obteniendo unos resultados inesperados.

A.3. Copia y configuración de los ficheros necesarios.

A.3.1. Copia de los archivos.

Una vez esta todo funcionando en orden, es decir, se puede visualizar cualquier página web con nuestro servidor desde otro cliente, hay que colocar las propias páginas del sistema en sus sitio.

[1] Todo esto se puede encontrar en la web oficial del Apache: Apache-Doc `http://httpd.apache.org/docs/2.0/`. Existe un pequeño resumen de instalación al final de este texto, en el capítulo A.6.

[2] El administrador del sistema tiene que tener mucho cuidado si aloja otras páginas web para no darles a ellos permisos de lectura, escrita y ejecución en sus scripts y provocar así un agujero en su seguridad. Por ello, es aconsejable que esta máquina sea un servidor dedicado exclusivamente al sistema TocToc.

1. Mover la carpeta *web_ TocToc* que contiene todos los ficheros PHP al directorio en el que el servidor Apache sirve las páginas web (denominado a partir de ahora "directorio *www*" o *"www"* simplemente.
2. Darle permisos de ejecución a los archivos para que PHP no tenga ningún problema con ellos. Eso significa que todos los archivos relacionados tienen que tener permisos de lectura para todos.
3. Guarda la carpeta *TocToc* que contiene los scripts de ejecución del sistema a tu equipo y dótale también de los permisos necesarios de ejecución. A lo largo del manual, nos referiremos a esta carpeta simplemente como *dir_ scripts*.

A.3.2. Configuración.

Los archivos que hay que configurar son los siguientes:

1. *www/include/colores.inc.*
2. *www/include/conexion.inc.*
3. *www/include/configuracion.inc.*
4. *www/include/TocToc.css*
5. *dir_ scripts/TocToc.sh.*
6. */Directorio_ apache/errors/error.php*

A continuación, se detalla la configuración y el uso de cada uno de ellos.

Archivo *colores.inc.* Este archivo guarda la configuración básica de los colores de la página. Permite cambiar el aspecto de todo el sistema, junto con el archivo *css* y el *xsl*. Los nombres de las variables son autoexplicativos, y el color tiene que escribirse en formato RGB. La imagen que se encuentra aquí, corresponde con la imagen que aparecerá de fondo en cada página web.

Archivo *conexion.inc* En este archivo se especifica, entre otras cosas, el *host* que contiene las bases de datos del sistema, el usuario que tiene acceso a las mismas y el *password* necesario para ello. Aquí tienes que poner los datos dependiendo de la instalación de MySQL en tu sistema.

Las dos constantes *BD_ USUARIOS* y *BD* indican el nombre de las dos bases de datos del sistema. Por lo que no es necesario cambiarlos salvo que se haga lo mismo en las bases de datos MySQL.

Archivo *configuracion.inc* Este archivo contiene la configuración específica de su sistema. Aquí se deben de introducir tanto algunos datos propios de su equipo, como otros datos que ya dependen de los resultados que se deseen que devuelva el sistema. A continuación se muestra el significado de las constantes aquí definidas:

1. *TOCTOC_ FOLDER*: indica la carpeta que contiene todas las páginas web del fichero. Tiene que tener la estructura *www/TOCTOC_ FOLDER/*.php*

2. *PAGINA_INTERNET*: a que página se direccionará el usuario cuando accede a Internet si previamente no ha elegido una.

3. *DIR_CLIENTES*: la carpeta que contiene los archivos que se utilizan para controlar los clientes. Tienen que ser la misma que la indicada en el script *TocToc.sh*.

4. *CITYTICKET_TOTAL*: aquí se indica el número de *citytickets* que quiere que maneje el sistema. La configuración de lo que se quiere que haga cada uno de ellos, viene definido en el vector *array_velocidad_conexion-[cityticket][0 (bajada) - 1 (subida)]*. Por ejemplo: para la configuración *array_velocidad_conexion[2][0]=16* significa que para el Cityticket 1, su velocidad de bajada es de 16KB/s.

5. *LOG*: esta constante contiene el nombre del fichero de logs del sistema.

6. *TIEMPO_LOG*: el tiempo en segundos que se mantienen las entradas en el *log* antes de trasladarlas a la copia de seguridad del mismo. Por defecto es un mes.

7. *LOG_BAK*: el nombre en donde se guardarán las entradas del *log* una vez pasen el tiempo máximo que deben de quedar registradas.

8. *DIRECTORIO_LOG*: el directorio donde se almacenará el *log* del sistema.

9. *REFRESCO_ACCESO_WEB*: aquí se indica el tiempo que tardará en refrescarse la ventana de control de Internet que se le abrirá al cliente. Debe ser menor que el tiempo indicado en *TocToc.sh* o se corre el riesgo de que se desconecten los clientes sin motivo alguno.

10. *RED_WIRELESS*: el nombre que le da Linux al dispositivo de acceso inalámbrico. Generalmente suele ser o *eth1* o *wlan0*.

11. *RED_INTERNET*: el nombre del dispositivo que sale a Internet. Generalmente será *eth0*.

12. *IP_GATEWAY*: la IP que tiene el ordenador que actúa como Gateway (en donde está instalado el sistema TocToc). A esta IP se dirigirán los clientes para salir a Internet.

13. *PUERTOS_ACEPTADOS*: el rango de puertos que se reserva el sistema para poder añadir aquellos servidores permitidos a los que los clientes pueden salir sin pagar el acceso a Internet.

14. *TC*: la dirección del ejecutable del programa *tc* de Linux.

15. *IPT*: la dirección del ejecutable del *iptables* de Linux.

16. *TOCTOC_SCRIPT_PATH*: en donde se almacenan los scripts del servidor TocToc.

17. *TOCTOC_SCRIPT*: el nombre del script que activa el servidor TocToc.

18. *ROOT*: esta constante contiene el nombre del administrador del sistema que en principio no se mostrará en el mismo (es decir, será imposible de borrar, modificar y demás). Se puede usar esta cuenta para tener un administrador principal del sistema sin que le aparezca a los demás encargados del mantenimiento del mismo.

19. *PAGINA_PRINCIPAL*: la página a la que se dirigirá el sistema cuando un usuario acceda exitosamente en el sistema.

20. *CSS*: El archivo que contiene la hoja de estilos en cascada. Escribirlo mediante una variable permite probar con varios diseños cambiando esta variable.

21. *TITULO*: El título que aparecerá en la cabecera del navegador. Así se permite una fácil integración con otros sistemas.

22. *NOMBRE:* El nombre con el que se referenciará al sistema. El único valor de esto es mostrarlo en algunas páginas web.

Archivo *TocToc.css*. Este archivo es una hoja de estilos en cascada que utiliza el sistema para darle el formato a los títulos de las páginas, los menús y demás. Se puede modificar a gusto del administrador sin que varíe en algo el comportamiento del sistema.

Archivo *TocToc.sh*. Este es el script principal de TocToc. Al ejecutarlo prepara al equipo para recibir a los clientes según sus niveles de acceso. Es también el que impide que salgan a Internet sin permiso. Para configurarlo, hay que mirar las siguientes constantes:

1. *IPT*: Al igual que en el archivo *configuracion.inc* esta constante indica donde se encuentra el binario para ejecutar las *iptables*. En la distribución probada se encuentra en: */sbin/iptables*.

2. *TC*: Similar al anterior, pero para la ejecución del *tc*. Se encuentra en el equipo probado en: */sbin/tc*.

3. *ETLAN*: La red *ethernet* por la que se sale a Internet. Algo del tipo: *152.41.51.0/22*.

4. *WLAN*: La dirección de la red *wireless* por la que acceden los clientes. En este caso se ha elegido *192.168.1.0/24*.

5. *ETH*: El nombre que el sistema Linux da al dispositivo de red *ethernet*. Por ejemplo *eth0*.

6. *WIR*: El nombre del dispositivo wireless que le da el sistema operativo. Generalmente *eth1* o *wlan0*.

7. *PWLAN*: Esta constante contiene aquella parte de la dirección de red que es común entre todos los clientes. Sería el equivalente al aplicar la máscara a la red, pero que por razones de diseño, tenemos que indicar aquí explícitamente. En nuestro caso *192.168.1.*

8. *PUERTOS_ACEPTADOS*: Aquí se indican el rango de puertos reservados por el sistema para aquellas páginas web a las que se permita el acceso de los clientes aunque no tengan conexión a Internet permitido. Los que se incluyen en el archivo son *9400:9800*.

9. *IPS_ACEPTADAS*: Aquí se indica la IP de aquellos servidores a los que se quiera dar acceso sin que exista una conexión a Internet. Aquí se indica aquellos servidores que siempre estarán implícitos en el sistema y no se podrán modificar desde la consola de administración.

10. *DIR_PHP*: La ruta que permite al sistema encontrar todos los ficheros PHP. Algo como */www/TocToc*.

11. *DIR_CLIENTES*: Aquí se indica una carpeta en el que el servidor apache tendrá permisos para poder dejar todos los archivos de configuración que necesita para su correcto funcionamiento. Algo como */TocToc/clientes*.

12. *DIR_SCRIPTS*: En que directorio están los demás scripts que necesita ejecutar el servidor. Por ejemplo, */TocToc/scripts*.

13. *LOG*: Nombre del fichero en donde se guardan los *logs* del sistema. El nombre es indiferente mientras se mantenga en el sistema.

14. *DIR_LOG*: Allí donde se sitúa el archivo del *log*. Siguiendo las pautas anteriores, quedaría algo así como */TocToc/log*.

15. *TIEMPO*: El intervalo de tiempo que se espera el servidor para comprobar si el cliente ha caído o no. Se aconseja que sea unos segundos superior al tiempo indicado en la constante *REFRESCO_ACCESO_WEB* del archivo *configuracion.inc*.

16. TOCTOC: Esta variable indica el número final de la IP del ordenador en el que está alojado el servidor. Nótese que *PWLAN.TOCTOC* tienen que formar una dirección IP válida. En nuestro caso sería un *1*, que acabaría formando *192.168.1.1*.

17. *DHCPmin* y *DHCPmax*: Estas dos variables indican que rangos de direcciones del servidor DHCP se distribuyen entre los clientes que se conectan por red. *PWLAN.DHCPmin* y *PWLAN.DHCPmax* tienen que formar direcciones IP válidas.

18. *RATEUP* y *RATEDOWN*: Estas dos variables indican el ancho de banda máximo de subida y de bajada que utilizará el sistema para repartir entre los clientes. Lo aconsejable es que sea un 75 % más o menos del ancho de banda total.

Archivo */Directorio_apache/errors/error.php* Este archivo deberá buscarse en la carpeta *error* del sistema TocToc si todavía no ha sido copiado a la carpeta de errores de Apache. Una vez realizado la copia, habrá que configurarlo revisando el valor de las siguientes constantes:

1. *IP_GATEWAY*: Aquí, al igual que en los archivos anteriores, se tendrá que indicar la dirección IP del dispositivo inalámbrico del ordenador en el que está instalado el sistema.

2. *RED_WIRELESS*: Escribir la dirección de la red Wireless para diferenciar los accesos al sistema como clientes TocToc o no. Es decir, aquellos equipos que entran por esta red son los que tiene que atrapar el sistema hasta que se dan de alta en el sistema.

3. *TOCTOC_FOLDER*: La carpeta en donde están todos los archivos PHP del sistema. Tiene que contener el mismo valor que *DIR_PHP* descrito en el apartado anterior.

A.4. Ejecución del sistema.

Una vez ha sido instalado y configurado todo lo anterior, la forma de que el sistema se ponga en funcionamiento es la siguiente.

1. Comprobar que los dispositivos de red del servidor están activos y funcionando correctamente.

2. Asegurarse que los servidores Apache y DHCP están en funcionamiento.

3. Poner en marcha el servidor TocToc mediante la ejecución del script *TocToc.sh*. Esto se hace de la siguiente forma: *"sh TocToc <start/stop><0/1>"* por ejemplo *"sh TocToc.sh start 0"*. El 0 indica que queremos que nos muestre por pantalla todos los mensajes, el 1 que los omita. Otra forma de poner en marcha el sistema es conectarse a TocToc `http://localhost/TocToc` e introducir un usuario que tenga privilegios de administrador en el sistema. Si es así, le aparecerá en el menú principal una opción de activación del sistema.

4. Ahora solo falta coger un cliente, conectarse a la red del sistema, y comprobar que todo funciona.

A.5. Detención del sistema.

También hay que saber como detener el sistema correctamente. Para detener el sistema, lo que hay que hacer es ejecutar el *script* descrito en la sección anterior de la siguiente forma: *"sh TocToc stop"*. Si no se detiene el sistema de esta forma, se queda un proceso activo en el sistema. Este proceso es el que vigila que no se queden clientes activos si estos han caído de forma inesperada. Tener este proceso activo no perjudica a la computadora, pero cuando se vuelva a ejecutar el sistema TocToc, el tener dos procesos que eliminan clientes inactivos puede ser problemático para el correcto funcionamiento del mismo. Si esto ocurre, es posible que clientes que no están inactivos sean eliminados por el sistema debido al funcionamiento doble de este proceso.

A.6. Instalación básica de los programas.

A.6.1. Instalación del OpenSSL y generación de un certificado propio.

La instalación del sistema OpenSSL depende del sistema Linux elegido. Existen paquetes binarios relativos a cada distribución que pueden ser usados, faci-

litando la tarea de instalación. De otra forma, se tendrá que descargar el código de la web oficial OpenSSL `http://www.openssl.org/` y compilarlo para su funcionamiento:

1. Descomprimir el paquete descargado: *"tar zxvf openssl-0.9.X.tar.gz"*.
2. Un simple *"./configure && make && make install"* debería bastar.

Una vez ha sido instalado, se debería pasar crear un certificado de seguridad. Para ello hay que seguir estos pasos:

1. Creación de la clave del servidor de 1024 bits: *"openssl genrsa -out ./server.key 1024"*.
2. Creación de la petición de certificado: *"openssl req -new -key server.key -out server.csr"*. Responder a las preguntas que se formulan. La contraseña hay que dejarla en blanco.
3. Generación de la clave privada: *"openssl x509 -req -days 365 -set_serial N -in server.csr -signkey server.key -out server.crt"*. En donde N es un número que se incrementa por el usuario por cada certificado creado.
4. Se pueden combinar ambos certificados en uno para mayor comodidad en su manejo: *"cat server.key server.crt >combined.pem"*.
5. También podemos comprobar que todo se ha generado correctamente de la siguiente forma: *"openssl x509 -in combined.pem -noout -text"*.
6. Si todo esta correcto, conviene dejar los certificados en su directorio histórico. Ese directorio es */etc/ssl/certs*.

A.6.2. Instalación básica de Apache.

A continuación se presentan los pasos básicos para instalar el servidor Apache en un computador. No se aconseja usar los paquetes que vienen con las distribuciones ya que no se pueden configurar con las opciones necesarias antes de instalarlo.

1. Lo primero que hay que hacer es bajarse el paquete con el fuente del Apache de la web oficial. Para ello, hay que dirigirse a: *Apache* `http://httpd.apache.org/download.cgi` y seleccionar la versión que se quiera bajar. El sistema TocToc a sido probado en la versión 2.0.55 de Apache.
2. Una vez se ha descargado el paquete, el siguiente paso es descomprimirlo. Para ello escribir en la línea de comandos: *"tar zxvf httpd-2_0_NN.tar.gz"*
3. Entrar en la carpeta creada httpd-2_0_NN y configurar el Apache con las siguientes opciones activas: *"./configure –prefix=/Apache –enable-so –enable-deflate –enable-ssl"*.
4. Instalar el Apache: *make && make install*

Si no ha habido ningún error durante la instalación del programa, el servidor debe de estar listo para ser usado. Ahora solo falta editar el archivo *"Directorio_apache/conf/httpd.conf"* y introducir unos cambios básicos para que reconozca nuestros documentos.

1. Buscar la sección *Dynamic Shared Object (DSO) Support* y añadir la siguiente línea: *"LoadModule php5_ module modules/libphp5.so"* que permite cargar PHP.

2. Buscar la cadena *Addtype* y añadir al final del listado de tipos insertados los documentos PHP de la forma: *"AddType application/x-httpd-php .php .phtml"* y *"AddType application/x-httpd-php-source .phps"*. Esto permite a Apache reconocer los documentos a partir de la terminación del archivo.

3. Buscar el string *Documentroot* y colocar aquí el directorio donde vayamos a albergar nuestras páginas web. Después hay que modificar unas líneas más abajo la sentencia *<Directory "/www/">* de forma que en *www* se indique el directorio seleccionado en *Documentroot*.

4. Añadir las páginas index.php a las páginas por defecto del servidor. Esto se consigue buscando la etiqueta *DirectoryIndex* y dejándola así: *"DirectoryIndex index.html index.html.var index.php index.htm"*. El orden es influyente, ya que indica al servidor la prioridad de las páginas a buscar.

5. Asegurarse que el usuario del servidor, es el usuario *apache*[3]. Si este usuario no existe en el sistema, añadirlo a los usuarios de Linux.

6. Generar el archivo */var/log/httpd/ssl_ request_ log.*

7. Para arrancar el apache en modo seguro utilizar: *"Directorio_ apache/bin/-apachectl startssl"*

A.6.3. Instalación de MySQL.

Para instalar MySQL, hay que dirigirse a MySQL `http://www.mysql.com/` y bajarse tanto los paquetes del servidor como del cliente. Existen paquetes para cada distribución de Linux, por lo que dependiendo de la que se tenga en la máquina, se deber seguir unas instrucciones u otras. En la página web está toda la información necesaria para ello. Por otra parte, existen dos aplicaciones que pueden resultar muy prácticas para gestionar las bases de datos. Sus nombres son: MySQL Query Browser y MySQL Administrator.

Una vez instalado el servidor, conviene añadir un password al administrador por razones de seguridad. Para ello, como root escribir: *"mysqladmin password el_ password_ que_ quieras"*.

Ahora lo que hay que hacer es introducir los esquemas de la base de datos que usará el sistema TocToc. Para ello importa el archivo de bases de datos que se adjunta con el sistema. Para ello, puedes usar un programa como el mysql-admin o hacerlo desde la propia consola de la siguiente forma:

1. Genera la base de datos correspondiente (como root): "*mysqladmin create toctoc_ database -p*".

2. Incluirle los esquemas que vienen con el paquete: "*mysql toctoc_ database <toctoc_ database.schema -p*".

[3]Técnicamente, aquí puede ir cualquier usuario salvo el *root*. Sin embargo, los scripts que se reparten con el sistema TocToc, están configurados específicamente con este usuario.

3. Repetir los pasos 1 y 2 para la base de datos que gestiona los usuarios: "*mysqladmin create toctoc_users -p*" y "*mysql toctoc_users <toctoc_users.schema -p*".

Después hay que hacer que la base de datos pertenezca al usuario *TocToc* para que el sistema pueda hacer las peticiones necesarias a la base de datos. Esto se hace en dos pasos. Primero entramos en mysql: *"mysql -u root -p"* para después cambiar los privilegios del usuario en la base de datos: *"grant all on TocToc_database.* to TocToc@localhost identified by 'el_password_que_queramos';" "grant all on TocToc_usuarios.* to TocToc@localhost identified by 'el_password_de_antes';"* y aplicamos lo hecho *"flush privileges"*.

Por último, hay que señalar que el esquema de la base de datos que se reparte con el resto de los ficheros, incluye ya un usuario generado con privilegios de administrador. El nombre es *root* y la contraseña de acceso es *toctocadmin*. Una vez testeado el sistema y comprobado que funciona, conviene cambiar lo antes posible la contraseña de este usuario. Este usuario es especial en tanto que no aparece en ningún sitio de administrador, por lo que conviene mantenerlo a salvo.

A.6.4. Instalación de PHP.

En esta sección se describe la instalación básica de PHP.

1. Configuración de la instalación de PHP para que use Apache y MySQL: *"./configure –with-apxs2=/Apache/bin/apxs –with-mysql"*.
2. Compilación e instalación: *"make && make install"*
3. Configuración de las opciones de PHP. Lo más cómodo es trabajar con el archivo de ejemplo que trae el propio paquete. Para ello *"cp php.ini-dist /usr/local/lib/php.ini"*.
4. Realizar las siguientes modificaciones en el php.ini anterior. Primero: En la cadena *doc_root* poner la dirección de la carpetas donde se guardan las páginas web a mostrar. Indicar en la cadena *session.save_path* la carpeta en donde se desea que se guarden las sesiones web. Tiene que ser una carpeta con los permisos adecuados para que se puedan depositar los archivos de sesión, por ejemplo la carpeta *"/tmp"*.

A.6.5. Instalación del servidor DHCP.

Aquí se muestran los pasos y toda la información necesaria para configurar un servidor DHCP.

1. Bájese el paquete o fuente del servidor de tu distribuidora Linux, de su proveedor de paquetes habitual o de esta dirección: FTPDHCP `ftp://ftp.phystech.com/pub/`
2. Si es un código fuente, compílelo siguiendo el procedimiento normal.
3. Generar el archivo */etc/dhcpd.conf* si este no existe. Generar la configuración del mismo dependiendo del tipo de red. Un ejemplo de como tiene que quedar esta red se encuentra en la siguiente dirección: *DHCP* `http://www.tldp.org/HOWTO/DHCP/x369.html`.

4. Añadir la siguiente entrada al encaminamiento: *"route add -host 255.255.-255.255 dev eth0"* siendo eth0 el dispositivo de red que permite el acceso a Internet.

5. Generar el archivo siguiente: *"touch /var/state/dhcp/dhcpd.leases"*.

6. Ejecutar el servidor: *"/usr/sbin/dhcpd"*.

A.6.6. Permisos de las carpetas.

Los permisos de las carpetas que contienen los archivos del sistema, tienen que dar acceso tanto al servidor Apache, como a aquel que tenga que ejecutar los scripts. Unos permisos inadecuados, puede resultar en que no funcione parte del sistema, o que este tenga un comportamiento inadecuado. Simplemente hay que asegurarse que todas las carpetas usadas en el proyecto tengan permiso de lectura y ejecución para el usuario *apache*. Hay que evitar que estos permisos afecten a cualquier otro usuario no autorizado.

En la carpeta donde se escriben los archivos de control de los clientes y en el de log, además tienen que tener los permisos de escritura adecuados.

Apéndice B

Manual del Programador.

B.1. Introducción.

Este capítulo tiene como finalidad ayudar a un programador a ampliar la arquitectura Toc-Toc. Pretende ser una guía para aquel que quiera continuar el desarrollo del mismo, explicando detalladamente que partes de la arquitectura se encargan de cada función del sistema.

B.2. Estructura general del sistema.

El sistema esta dividido en tres partes: Un conjunto de *scripts* que controlan el funcionamiento del servidor, una interfaz visual generada mediante distintos lenguajes de desarrollo web y un sistema de almacenamiento de datos, compuestos por dos bases de datos.

B.2.1. Estructura de los *scripts*.

Aunque no es necesario, los distintos *scripts* y archivos que genera el sistema están dividido en diversas carpetas para una mayor claridad. Estos *scripts* se encargan del funcionamiento interno del servidor y del control de los usuarios. Están divididos por defecto en:

- TocToc_scripts: contiene los scripts principales del sistema, junto con el archivo *starstop.txt* que indica si el sistema está o no activo.
- TocToc_log: contiene el *log* del sistema y también la copia de aquellas entradas anteriores a dos meses.
- TocToc_clientes: aquí el sistema genera los archivos de control de cada uno de los clientes, indicando en cada caso si están o no activos en el sistema y han salido a Internet.

B.2.2. Estructura de la interfaz web.

La interfaz web esta contenida en la carpeta que sirve el servidor Apache. A esta carpeta la llamaremos carpeta raíz de la interfaz web. La jerarquía de los ficheros de esta estructura queda representada en la figura B.2.

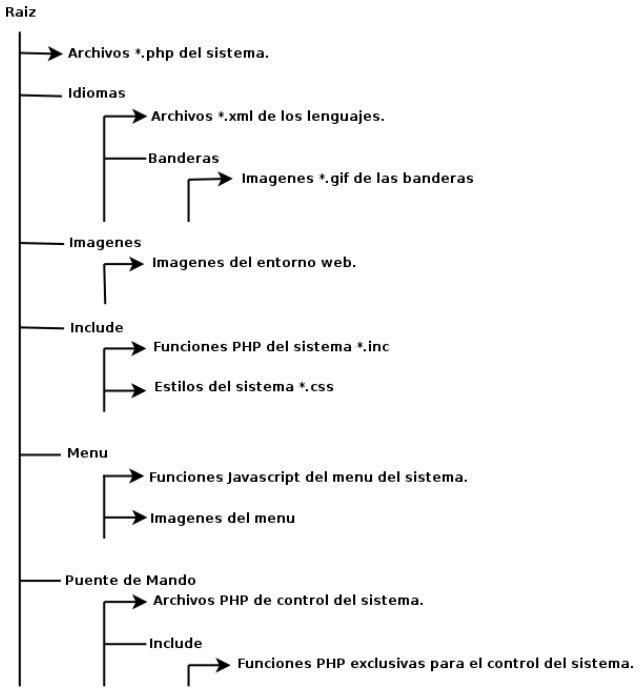

Figura B.1: Estructura de la interfaz web.

Como se puede ver en dicha figura, se distribuye de la siguiente forma:

1. Los archivos principales de la web están depositados en esta carpeta raíz directamente. Como en todas las webs, el archivo inicial será aquel llamado *index.php*.

2. La carpeta "Idiomas" contiene todos los archivos utilizados en el multilingüismo. Estos archivos son los XML que contienen las diversas traducciones de la interfaz web.

   a) La carpeta "Banderas" contiene todas las banderas que permitirán a un usuario escoger idioma.

3. "Imagenes" es la carpeta que contiene todas las imágenes del sistema relativas a su diseño. En el caso más simple, solamente contiene el logo del sistema.

4. La carpeta "Include" contiene todas las funciones PHP que utilizan el resto de archivos para mantener el control del sistema. Dentro de esta carpeta cabe destacar los siguientes archivos:

   a) FuncionesPHP.inc: contiene todas las funciones de sesiones, estructuras de datos, etc.

   b) FuncionesBD.inc: incluye todas las funciones de acceso a la base de datos del sistema.

c) FuncionesBD_USUARIO.inc: aquí están todas las funciones que acceden a la base de datos para el control de usuarios.

d) XML.inc: incluye todas las funciones que permiten el multilingüismo y generan el XML que después se enviará al cliente.

e) RedWireless.inc: Todas las funciones de acceso a Internet del sistema. Incluye el registro del *log* de usuarios.

5. "Menu" es la carpeta donde se guardan todos los archivos relativos al menú que permite a un usuario elegir las opciones disponibles en el sistema. Como esta generado en DHTML, aquí se incluyen las funciones Javascript que lo componen y las imágenes que aparecerán después en él.

6. "Puente de mando" es la carpeta que contiene todos los archivos de administración del sistema. La decisión de crear una carpeta aparte para esto reside en que así se permite darle algunos permisos más restrictivos si así se desea. Esta carpeta imita la estructura global, pero limitándola a la parte de control:

   a) Los archivos PHP de esta carpeta componen las páginas que permiten al administrador modificar el sistema.

   b) La carpeta "Include", al igual que su homóloga en el nivel superior, contiene las funciones PHP exclusivas para esto.

B.2.3. Estructura de las bases de datos.

El sistema esta compuesto por dos bases de datos. La primera de ellas, llamada *toctoc_database* contiene todos los datos relativos al sistema. La segunda, llamada *toctoc_users* contiene todas las tablas de los usuarios. El motivo de la división del sistema en dos bases de datos, consiste en proporcionar una mayor seguridad a los datos y facilitar la tarea al programador para la ampliación del sistema sin riesgo a perder los usuarios dados ya de alta en el mismo.

B.3. Ampliación del sistema.

En los puntos sucesivos, se pretende explicar que estrategia hay que seguir para ampliar el sistema.

B.3.1. Ampliación del menú principal.

Por menú principal se hace referencia al menú que aparece una vez un usuario entra vía web en el sistema. Este menú enumera las distintas opciones a las que puede acceder un usuario, tales como la modificación de su perfil o el acceso a Internet. La estructura de este menú fue generado por un programa especializado en la generación de menús dinámicos en lenguaje DHTML. Sin embargo, como el sistema está basado en PHP, las entradas tienen que ser realizadas a mano.

La estructura descrita por el programa define un objeto principal que contiene alguna información general para todas las opciones, como las imágenes que indican que existe un submenú. La primera entrada del menú debe contener, además, la información del formato y colores visuales de cada opción del menú:

- La primera línea de todas, indica el comienzo del menú con el objeto principal. Esta línea es la siguiente:

  stm_bm(["menu4bb3", 430, "<?php echo MENU_NO_IMG_1 ?>", "<?php echo MENU_IMG_2 ?>", 0, "", "", 0, 0, 250, 0, 1000, 1, 0, 0, "", "", 0], this);

  Esta línea contiene el nombre del menú y la estructura genérica del mismo. Aquí se indican las imágenes que se usan cuando es un menú desplegable o no lo es. Las variable *MENU_IMG_2* y MENU_IMG_2 indican que imágenes se tienen que mostrar al lado del texto (cuando está seleccionado y no seleccionado respectivamente). Ambas están definidas en la página *colores.inc*.

- La última línea debe de todo el menú debe de ser:

  stm_em();

- La primera línea de los campos del menú contiene el formato. Es decir, los colores del mismo. Esta línea es:

  stm_bp("p0", [1, 4, 0, 0, 2, 3, 0, 7, 100, "", -2, "", -2, 50, 0, 0, $COLOR1, $COLOR2, "", 3, 1, 1, $COLOR_RECUADRO_MENU]);

  El primer campo indica que es el objeto inicial del menú, del que colgarán las demás opciones del mismo. Las variables *$COLORX* y *$COLOR_RECUADRO_MENU*, contienen los colores que se utilizarán en la creación del menú. Los colores, como es el campo principal, se heredarán para el resto de opciones.

- La siguiente líneas es ya la entrada que aparecerá visualmente en la página:

  stm_ai("p0i0", [0, $TEXTO, "", "", -1, -1, 0, $LINK, $DESTINO, "", "", "", "", 0, 0, 0, "", "", 0, 0, 0, 1, 2, $COLOR_MENU, 0, $COLOR_TEXTO_MENU, 0, "", "", 3, 3, 1, 4, "$COLOR_LINK", $COLOR_ALINK, $COLOR_VLINK, $COLOR_TEXTO_SELEC", "bold 10pt Verdana", "bold 10pt Verdana", 0, 0]);

  Esta entrada del menú se ha llamado "*p0i0*". Mostrará el texto, que aparezca en $TEXTO. El campo $LINK indica hacia donde se navegará al pulsar este enlace. El campo relleno con $DESTINO ("*_self*", "_parent", "_blank") indica donde se abrirá el enlace al igual que en HTML. Las variables *COLOR_X* contienen los diversos colores con los que se mostrará el menú. Los dos campos siguientes indican el tamaño de letra y el tipo. La repetición de la información es debido a que permite modificar el tipo de letra según si la opción está seleccionada o no.

- Una vez se ha definido el primer botón del menú, todos los botones que aparecen detrás de este heredan las opciones que no se definan. Por ejemplo:

  stm_aix("p0i2", "p0i0", [0, $TEXTO, "", "", -1, -1, 0, $LINK]);

Significa que la siguiente opción, llamada "*p0i2*" está al mismo nivel que la "*p0i0*" (por tanto tendrán los mismos colores y tipos de letra. Aparecerá el texto *$TEXTO* en el botón del menú y al pulsarlo, el navegador irá a *$LINK*. Las opciones no definidas las hereda del objeto "*p0i0*".

- Para generar un submenú, hay que ampliar los campos a rellenar añadiendo las opciones que indican que se muestre la imagen que representa la presencia de un submenú (en este caso el dibujo de una flecha).

  stm_aix("p0i3", "p0i0", [0, $TEXTO, "", "", -1, -1, 0, $URL, $DESTINO, "", "", "", "", 0, 0, 0, "menu/arrow_rw.gif", "menu/arrow_rw.gif", 7, 7]);

  En este caso, las variables $URL y $DESTINO suelen estar vacías. Inmediatamente después de esto, se incluye esta instrucción que indica que las siguientes opciones pertenencen a este submenú.

  stm_bpx("p1", "p0", [1, 2, 0, 0, 2, 3, 0, 0]);

  Y se van incluyendo las nuevas opciones del menú:

  stm_aix("p1iX", "p0i0", [0, $TEXTO, "", "", -1, -1, 0, $URL]);

  La X representa un número creciente que es el identificador de la siguiente opción.

  Y se finaliza el menú:

  stm_ep(); stm_ep();

  De esta forma, todas las opciones incluidas dentro de las instrucciones de inicio y fin del submenú aparecerán solamente cuando el usuario selecciona la opción adecuada. Esto debe hacerse para cada submenú.

  En el Ejemplo 13, se muestra la creación de un menú completo.

B.3.2. Añadir un nuevo campo para los usuarios.

Si se quiere añadir alguna nueva opción que se debe de guardar en la base de datos de los usuarios, lo primero sería obtener los datos del mismo. Esto se puede hacer en las páginas *registrarse.php y perfil.php* que es donde un usuario rellena las opciones del mismo. Para ello hay que modificar el formulario y añadir los cambios pertinentes. Para que estos datos se guarden en la base de datos, habrá que modificar las páginas *crea_usuario.php* y *actualiza_usuario.php* y modificar las funciones PHP incluidas para obtener y guardar los campos en la base de datos.

Posiblemente sea interesante mantener esos datos en la sesión de usuario, para utilizarlos más tarde. Para ello, hay que revisar las funciones *lee_sesion y guarda_sesion* y añadir los campos adecuados.

Algorithm 11 Ejemplo de un menú.

```
stm_bm(["menu4bb3", 430, "", "", 0, "", "", 0, 0, 250, 0, 1000, 1, 0, 0, "", "",
0], this);
stm_bp("p0", [1, 4, 0, 0, 2, 3, 0, 7, 100, "", -2, "", -2, 50, 0, 0, "#000000",
"#000000", "", 3, 1, 1, "#FFFFFF"]);

stm_ai("p0i0", [0, "Opcion1", "", "", -1, -1, 0, "opcion1.html", "_self", "", "",
"", "", 0, 0, 0, "", "", 0, 0, 0, 1, 2, "#000000", 0, "#FFFFFF", 0, "", "", 3,
3, 1, 4, "#FFFFFF", "#CCCCCC", "#FFFFFF", "#CCCCCC", "bold 10pt
Verdana", "bold 10pt Verdana", 0, 0]);

stm_aix("p0i1", "p0i0", [0, "Submenu2", "", "", -1, -1, 0, "", "", "", "", "", "",
0, 0, 0, "flecha.gif", "flecha.gif", 7, 7]);

 stm_bpx("p1", "p0", [1, 2, 0, 0, 2, 3, 0, 0]);
  stm_aix("p1i1", "p0i0", [0, "Opcion21", "", "", -1, -1, 0, "opcion21.html"]);
  stm_aix("p1i2", "p0i0", [0, "Opcion22", "", "", -1, -1, 0, "opcion22.html"]);
  stm_aix("p1i3", "p0i0", [0, "Opcion23", "", "", -1, -1, 0, "opcion23.html"]);
 stm_ep();

stm_ep();
stm_em();
```

B.3.3. Multilingüismo.

Modificación del XML. El multilingüismo esta basado en la utilización de un formato XML que permite la diferenciación de los idiomas que componen los párrafos. Este XML contiene una estructura que asocia a una etiqueta determinada el significado correspondiente a cada uno de los idiomas. Por ejemplo, en el Algoritmo 14 se muestra el ejemplo más simple de XML aplicado al multilingüismo.

Algorithm 12 Ejemplo de una página XML multilingüe.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<toctoc>
  //Etiqueta de párrafo.
  <hola_mundo>
    //Etiquetas de idioma junto con el significado.
    <esp>Hola mundo</esp>
    <eng>Hello Word</eng>
  </hola_mundo>
</toctoc>
```

La ampliación de una página ya creada, resulta sencilla gracias a este formato. Se realiza dependiendo de como se quiera ampliar:

- Si la ampliación es un nuevo idioma, solamente hay que añadir entre dos nuevas etiquetas que lo diferencien, la traducción correspondiente.
- Si se quiere añadir un nuevo párrafo, habrá que crear una etiqueta de

párrafo que lo diferencie. Dentro habrá que escribir las diferentes versiones en cada uno de los idiomas deseados.

Modificación de los archivos PHP. Los archivos de PHP se encargan de mostrar el contenido adecuado que se recoge en los archivos XML. Para ello, PHP tiene que hacer dos pasos:

- Abrir el archivo XML adecuado y obtener el texto. Esto de hace mediante:

  include("include/XML.inc");
  $fichero="idiomas/ejemplo.xml";
  abreXML($fichero,$arr_vals, $arr_index);

  La primera línea incluye las funciones que se usan más tarde. La variable *$fichero* contiene la ruta al XML relacionado. La función *abreXML()* abre el fichero indicado y guarda el XML en forma de matriz dentro de la variable *$arr_vals* que será utilizada más tarde.

- Imprimir el párrafo adecuado en el idioma deseado. Esto se hace con la función:

  obtieneCampoXML($arr_vals, $PARRAFO, $IDIOMA, $SALIDA); echo $SALIDA;

  Esta función obtiene el párrafo etiquetado por la variable *$PARRAFO*, escogiendo el idioma *$IDIOMA* y lo guarda en la variable *$SALIDA*. Imprimir esta última variable mostrará el texto.

Estos pasos tendrán que repetirse para cada página PHP que contengan texto y por cada uno de los párrafos escritos.

Añadir un nuevo idioma. Para añadir un nuevo idioma, aparte de las traducciones pertinentes en cada página, hay que añadir una nueva bandera en la página principal. Cada bandera recarga la página enviando una variable que contiene el código del idioma, que es el que se utilizará a su vez en el XML para diferenciarlos.

B.3.4. Conexión a Internet.

La conexión a Internet viene definida dentro de la página *redWireless.inc* que contiene todas las funciones necesarias para su manejo. Primero se presenta una explicación de cada una de ellas:

- *escribe_log*(): guarda una cadena dentro del log del sistema.
- *reduce_log*(): deja en el log solamente aquellas entradas que tienen menos de un periodo de tiempo definido por la constante *TIEMPO_LOG*. El resto de entradas se guardan en la copia de seguridad del log.
- *da_internet_usuario*(): se encarga de generar las colas de TC específicas para el cliente que invoca a esta función. De esta manera se permite el acceso de este a Internet.

- *mantiene_internet_usuario*(): esta función se encarga de que el ordenador principal no dé de baja al cliente que la invoca. Debe ser llamada periódicamente o el sistema pensará que el cliente a perdido su conexión.
- *obtiene_estado_cliente*(): obtiene del sistema el estado del cliente, es decir, comprueba si el sistema da por conectado o desconectado este cliente.
- *obtiene_estado_sistema*(): indica si TocToc está encendido o apagado.
- *cierra_internet_usuario*(): cierra correctamente la conexión de un usuario.

De todas estas funciones la más importante es *da_internet_usuario*(). Esta es la función que hay que revisar si se pretende cambiar la funcionalidad de la conexión a Internet. Aquí habrá que indicar la nueva cola si se quiere usar algún otro tipo de cola o es donde se añadirá el IMQ en caso de que se quiera utilizar.

B.3.5. Generación de la ayuda del sistema.

La generación de la página de ayuda viene facilitada por dos funciones de PHP. Estas funciones se encargan de formar la estructura del índice de ayuda, los enlaces entre el índice y la explicación, y la propia explicación. Estas funciones:

- *ayudamenu($arr_vals, $ANCLA, $TEXTO)*: Esta función genera el menú de ayuda, mostrando el texto indicado en la página XML por la etiqueta incluida en la variable *$TEXTO* y que llevará al párrafo que tenga el enlace *$ANCLA*. La variable *$arr_vals* es, como se ha explicado anteriormente, la que contiene el texto obtenido de la página XML.
- *ayudaexplicacion($arr_vals, $TITULO, $EXPLICACION, $ANCLA)*: genera el párrafo explicativo al que se dirigirá el enlace del menú con el enlace *$ANCLA* y que se compondrá por un título y su explicación compuestos por el texto que devuelva el XML a pasarle las variables *$TITULO* y *$EXPLICACION* respectivamente.

Por tanto, para cada entrada en la ayuda habrá que incluir cada una de las funciones anteriores, generando así un índice principal y una explicación que aparecerá más abajo.

B.3.6. Ampliación general de las páginas PHP.

Por motivos de facilidad de desarrollo futuro, cada página que se utiliza en el sistema incluyen las páginas *cabecera.inc* y *pie.inc*. La primera de ellas se importa justo al principio del documento, mientras que la segunda se incluye justo al final. La finalidad de estas dos páginas es la aplicación de un elemento común en todas las páginas de forma sencilla. Por ejemplo, en *cabecera.inc* se incluyen todos los META-TAGs que se utilizan en todas las páginas, de forma que el mantenimiento de estos resulta muy sencillo en todas las páginas. El archivo *pie.inc* está en este momento vacío y se puede utilizar para añadir elementos al final de la página, como por ejemplo, contadores de visitas y otros elementos comunes como logotipos.

B.3.7. Archivos de protección.

El arquitectura TocToc incluye algunas funciones de protección para que no existan accesos no autorizados en el sistema. Estas funciones están en los archivos *proteccion.inc* y *proteccion_internet.inc*. El primero de los dos archivos, contienen unas funciones que impiden el acceso a una página a aquellos navegadores que no hayan generado una sesión de PHP. Como la sesión solamente se genera si el usuario entra en el sistema introduciendo su nombre de usuario y su contraseña, solamente los usuarios dados de alta pueden tener acceso al sistema. El segundo archivo, *proteccion_internet.inc*, es similar al primero salvo que impide el acceso a ciertas páginas a los usuarios que no tienen permitida la conexión a Internet.Introducción.

Este documento tiene como finalidad ayudar a un programador a ampliar la arquitectura Toc-Toc. Pretende ser una guía para aquel que quiera continuar el desarrollo del mismo, explicando detalladamente que partes de la arquitectura se encargan de cada función del sistema.

B.4. Estructura general del sistema.

El sistema esta dividido en tres partes: Un conjunto de *scripts* que controlan el funcionamiento del servidor, una interfaz visual generada mediante distintos lenguajes de desarrollo web y un sistema de almacenamiento de datos, compuestos por dos bases de datos.

B.4.1. Estructura de los *scripts*.

Aunque no es necesario, los distintos *scripts* y archivos que genera el sistema están dividido en diversas carpetas para una mayor claridad. Estos *scripts* se encargan del funcionamiento interno del servidor y del control de los usuarios. Están divididos por defecto en:

- TocToc_scripts: contiene los scripts principales del sistema, junto con el archivo *starstop.txt* que indica si el sistema está o no activo.
- TocToc_log: contiene el *log* del sistema y también la copia de aquellas entradas anteriores a dos meses.
- TocToc_clientes: aquí el sistema genera los archivos de control de cada uno de los clientes, indicando en cada caso si están o no activos en el sistema y han salido a Internet.

B.4.2. Estructura de la interfaz web.

La interfaz web esta contenida en la carpeta que sirve el servidor Apache. A esta carpeta la llamaremos carpeta raíz de la interfaz web. La jerarquía de los ficheros de esta estructura queda representada en la figura B.2.

Como se puede ver en dicha figura, se distribuye de la siguiente forma:

1. Los archivos principales de la web están depositados en esta carpeta raíz directamente. Como en todas las webs, el archivo inicial será aquel llamado *index.php*.

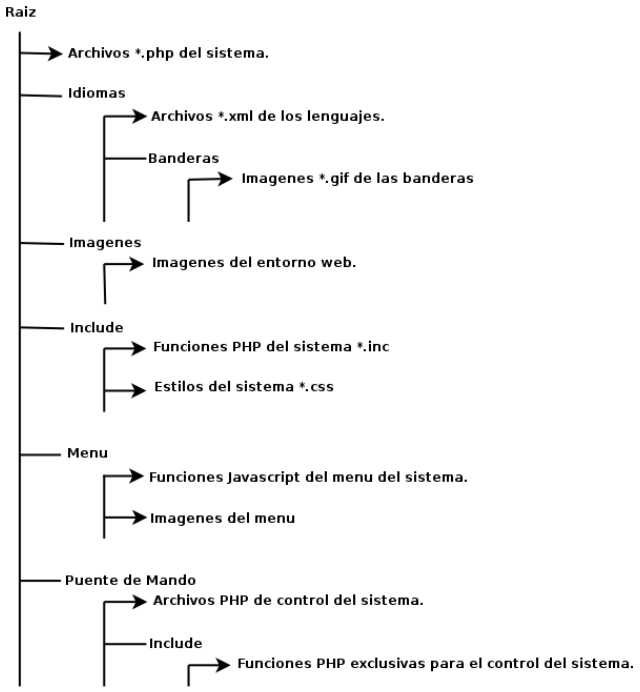

Figura B.2: Estructura de la interfaz web.

2. La carpeta "Idiomas" contiene todos los archivos utilizados en el multilingüismo. Estos archivos son los XML que contienen las diversas traducciones de la interfaz web.

   a) La carpeta "Banderas" contiene todas las banderas que permitirán a un usuario escoger idioma.

3. "Imagenes" es la carpeta que contiene todas las imágenes del sistema relativas a su diseño. En el caso más simple, solamente contiene el logo del sistema.

4. La carpeta "Include" contiene todas las funciones PHP que utilizan el resto de archivos para mantener el control del sistema. Dentro de esta carpeta cabe destacar los siguientes archivos:

   a) FuncionesPHP.inc: contiene todas las funciones de sesiones, estructuras de datos, etc.

   b) FuncionesBD.inc: incluye todas las funciones de acceso a la base de datos del sistema.

   c) FuncionesBD_USUARIO.inc: aquí están todas las funciones que acceden a la base de datos para el control de usuarios.

   d) XML.inc: incluye todas las funciones que permiten el multilingüismo y generan el XML que después se enviará al cliente.

e) RedWireless.inc: Todas las funciones de acceso a Internet del sistema. Incluye el registro del *log* de usuarios.

5. "Menu" es la carpeta donde se guardan todos los archivos relativos al menú que permite a un usuario elegir las opciones disponibles en el sistema. Como esta generado en DHTML, aquí se incluyen las funciones Javascript que lo componen y las imágenes que aparecerán después en él.

6. "Puente de mando" es la carpeta que contiene todos los archivos de administración del sistema. La decisión de crear una carpeta aparte para esto reside en que así se permite darle algunos permisos más restrictivos si así se desea. Esta carpeta imita la estructura global, pero limitándola a la parte de control:

   a) Los archivos PHP de esta carpeta componen las páginas que permiten al administrador modificar el sistema.

   b) La carpeta "Include", al igual que su homóloga en el nivel superior, contiene las funciones PHP exclusivas para esto.

B.4.3. Estructura de las bases de datos.

El sistema esta compuesto por dos bases de datos. La primera de ellas, llamada *toctoc_database* contiene todos los datos relativos al sistema. La segunda, llamada *toctoc_users* contiene todas las tablas de los usuarios. El motivo de la división del sistema en dos bases de datos, consiste en proporcionar una mayor seguridad a los datos y facilitar la tarea al programador para la ampliación del sistema sin riesgo a perder los usuarios dados ya de alta en el mismo. A continuación se presentan las tablas que componen la base de datos con una pequeña explicación.

B.5. Ampliación del sistema.

En los puntos sucesivos, se pretende explicar que estrategia hay que seguir para ampliar el sistema.

B.5.1. Ampliación del menú principal.

Por menú principal se hace referencia al menú que aparece una vez un usuario entra vía web en el sistema. Este menú enumera las distintas opciones a las que puede acceder un usuario, tales como la modificación de su perfil o el acceso a Internet. La estructura de este menú fue generado por un programa especializado en la generación de menús dinámicos en lenguaje DHTML. Sin embargo, como el sistema está basado en PHP, las entradas tienen que ser realizadas a mano.

La estructura descrita por el programa define un objeto principal que contiene alguna información general para todas las opciones, como las imágenes que indican que existe un submenú. La primera entrada del menú debe contener, además, la información del formato y colores visuales de cada opción del menú:

- La primera línea de todas, indica el comienzo del menú con el objeto principal. Esta línea es la siguiente:

Campo	Tipo	Significado
nick	VARCHAR(45)	El nombre de usuario.
pwd	VARCHAR(45)	La contraseña de acceso (encriptada).
cityticket	INTEGER	El nivel de acceso a Internet.
nombre	VARCHAR(45)	El nombre del cliente.
apellidos	VARCHAR(45)	Los apellidos del cliente.
pais	VARCHAR(45)	El país de procedencia del cliente.
email	VARCHAR(45)	Una dirección de contacto del cliente.
idioma	VARCHAR(10)	El idioma preferido de la página.
tmp_pwd	VARCHAR(45)	Contraseña temporal (encriptada).
tiempo_tmp_pwd	TIMESTAMP	Fecha de la contraseña temporal.
dispositivo	VARCHAR(15)	Clave ajena a la tabla dispositivos.
color	INTEGER	Clave ajena a la tabla colores.
resolucion	INTEGER	Clave ajena a la tabla resoluciones.
vel_conexion	INTEGER	Velocidad de descarga del cliente.
vel_subida	INTEGER	Velocidad de subida del cliente.
IP	VARCHAR(15)	IP que está usando el cliente .
hora_conexion	TIMESTAMP	Hora a la que ha iniciado la conexión.
admin	INT(1)	Los derechos de administración.
fecha_fin_conexion	TIMESTAMP	Hora de la ultima desconexión.

Cuadro B.1: Tabla de clientes.

Campo	Tipo	Significado
dispositivo	VARCHAR(15)	Los dispositivos válidos en el sistema.

Cuadro B.2: Tabla de dispositivos.

Campo	Tipo	Significado
dispositivo	VARCHAR(15)	Clave ajena a dispositivos.
tipo	VARCHAR(15)	El nombre de la resolución.
id	INTEGER	Un identificador autoincrementable.

Cuadro B.3: Tabla de resoluciones.

Campo	Tipo	Significado
dispositivo	VARCHAR(15)	Clave ajena a dispositivos.
profundidad	VARCHAR(25)	El color aplicado.
id	INTEGER	Un identificador autoincrementable.

Cuadro B.4: Tabla de colores.

stm_bm(["menu4bb3", 430, "<?php echo MENU_NO_IMG_1 ?>", "<?php echo MENU_IMG_2 ?>", 0, "", "", 0, 0, 250, 0, 1000, 1, 0, 0, "", "", 0], this);

Campo	Tipo	Significado
nombre	VARCHAR(10)	Nombre del servidor
puerto	VARCHAR(5)	Puerto (si se gestiona por puertos).
direccion	VARCHAR(50)	URL
ip	VARCHAR(17)	Dirección IP del servidor.
dest_port	VARCHAR(5)	Puerto de destino del servidor.

Cuadro B.5: Tabla de servidores.

Campo	Tipo	Significado
idioma	VARCHAR(15)	Nombre del idioma.
bandera	VARCHAR (15)	Dirección a la imagen.
id	INTEGER	La etiqueta XML que se usará.

Cuadro B.6: Tabla de idiomas.

Esta línea contiene el nombre del menú y la estructura genérica del mismo. Aquí se indican las imágenes que se usan cuando es un menú desplegable o no lo es. Las variable *MENU_IMG_2* y MENU_IMG_2 indican que imágenes se tienen que mostrar al lado del texto (cuando está seleccionado y no seleccionado respectivamente). Ambas están definidas en la página *colores.inc*.

- La última línea debe de todo el menú debe de ser:

  stm_em();

- La primera línea de los campos del menú contiene el formato. Es decir, los colores del mismo. Esta línea es:

  stm_bp("p0", [1, 4, 0, 0, 2, 3, 0, 7, 100, "", -2, "", -2, 50, 0, 0, $COLOR1, $COLOR2, "", 3, 1, 1, $COLOR_RECUADRO_MENU]);

  El primer campo indica que es el objeto inicial del menú, del que colgarán las demás opciones del mismo. Las variables *$COLORX* y *$COLOR_RECUADRO_MENU*, contienen los colores que se utilizarán en la creación del menú. Los colores, como es el campo principal, se heredarán para el resto de opciones.

- La siguiente líneas es ya la entrada que aparecerá visualmente en la página:

  stm_ai("p0i0", [0, $TEXTO, "", "", -1, -1, 0, $LINK, $DESTINO, "", "", "", "", 0, 0, 0, "", "", 0, 0, 0, 1, 2, $COLOR_MENU, 0, $COLOR_TEXTO_MENU, 0, "", "", 3, 3, 1, 4, "$COLOR_LINK", $COLOR_ALINK, $COLOR_VLINK, $COLOR_TEXTO_SELEC", "bold 10pt Verdana", "bold 10pt Verdana", 0, 0]);

  Esta entrada del menú se ha llamado "*p0i0*". Mostrará el texto, que aparezca en $TEXTO. El campo $LINK indica hacia donde se navegará al pulsar este enlace. El campo relleno con $DESTINO ("*_self*",

"_parent", "_blank") indica donde se abrirá el enlace al igual que en HTML. Las variables *COLOR_X* contienen los diversos colores con los que se mostrará el menú. Los dos campos siguientes indican el tamaño de letra y el tipo. La repetición de la información es debido a que permite modificar el tipo de letra según si la opción está seleccionada o no.

- Una vez se ha definido el primer botón del menú, todos los botones que aparecen detrás de este heredan las opciones que no se definan. Por ejemplo:

  stm_aix("p0i2", "p0i0", [0, $TEXTO, "", "", -1, -1, 0, $LINK]);

  Significa que la siguiente opción, llamada "*p0i2*" está al mismo nivel que la "*p0i0*" (por tanto tendrán los mismos colores y tipos de letra. Aparecerá el texto *$TEXTO* en el botón del menú y al pulsarlo, el navegador irá a *$LINK*. Las opciones no definidas las hereda del objeto "*p0i0*".

- Para generar un submenú, hay que ampliar los campos a rellenar añadiendo las opciones que indican que se muestre la imagen que representa la presencia de un submenú (en este caso el dibujo de una flecha).

  stm_aix("p0i3", "p0i0", [0, $TEXTO, "", "", -1, -1, 0, $URL, $DESTINO, "", "", "", "", 0, 0, 0, "menu/arrow_rw.gif", "menu/arrow_rw.gif", 7, 7]);

  En este caso, las variables $URL y $DESTINO suelen estar vacías. Inmediatamente después de esto, se incluye esta instrucción que indica que las siguientes opciones pertenencen a este submenú.

  stm_bpx("p1", "p0", [1, 2, 0, 0, 2, 3, 0, 0]);

  Y se van incluyendo las nuevas opciones del menú:

  stm_aix("p1iX", "p0i0", [0, $TEXTO, "", "", -1, -1, 0, $URL]);

  La X representa un número creciente que es el identificador de la siguiente opción.

  Y se finaliza el menú:

  stm_ep(); stm_ep();

  De esta forma, todas las opciones incluidas dentro de las instrucciones de inicio y fin del submenú aparecerán solamente cuando el usuario selecciona la opción adecuada. Esto debe hacerse para cada submenú.

  En el Ejemplo 13, se muestra la creación de un menú completo.

Algorithm 13 Ejemplo de un menú.

```
stm_bm(["menu4bb3", 430, "", "", 0, "", "", 0, 0, 250, 0, 1000, 1, 0, 0, "", "",
0], this);
stm_bp("p0", [1, 4, 0, 0, 2, 3, 0, 7, 100, "", -2, "", -2, 50, 0, 0, "#000000",
"#000000", "", 3, 1, 1, "#FFFFFF"]);

stm_ai("p0i0", [0, "Opcion1", "", "", -1, -1, 0, "opcion1.html", "_self", "", "",
"", "", 0, 0, 0, "", "", 0, 0, 0, 1, 2, "#000000", 0, "#FFFFFF", 0, "", "", 3,
3, 1, 4, "#FFFFFF", "#CCCCCC", "#FFFFFF", "#CCCCCC", "bold 10pt
Verdana", "bold 10pt Verdana", 0, 0]);

stm_aix("p0i1", "p0i0", [0, "Submenu2", "", "", -1, -1, 0, "", "", "", "", "", "",
0, 0, 0, "flecha.gif", "flecha.gif", 7, 7]);

 stm_bpx("p1", "p0", [1, 2, 0, 0, 2, 3, 0, 0]);
  stm_aix("p1i1", "p0i0", [0, "Opcion21", "", "", -1, -1, 0, "opcion21.html"]);
  stm_aix("p1i2", "p0i0", [0, "Opcion22", "", "", -1, -1, 0, "opcion22.html"]);
  stm_aix("p1i3", "p0i0", [0, "Opcion23", "", "", -1, -1, 0, "opcion23.html"]);
 stm_ep();

stm_ep();
stm_em();
```

B.5.2. Añadir un nuevo campo para los usuarios.

Si se quiere añadir alguna nueva opción que se debe de guardar en la base de datos de los usuarios, lo primero sería obtener los datos del mismo. Esto se puede hacer en las páginas *registrarse.php y perfil.php* que es donde un usuario rellena las opciones del mismo. Para ello hay que modificar el formulario y añadir los cambios pertinentes. Para que estos datos se guarden en la base de datos, habrá que modificar las páginas *crea_usuario.php* y *actualiza_usuario.php* y modificar las funciones PHP incluidas para obtener y guardar los campos en la base de datos.

Posiblemente sea interesante mantener esos datos en la sesión de usuario, para utilizarlos más tarde. Para ello, hay que revisar las funciones *lee_sesion y guarda_sesion* y añadir los campos adecuados.

B.5.3. Multilingüismo.

Modificación del XML. El multilingüismo esta basado en la utilización de un formato XML que permite la diferenciación de los idiomas que componen los párrafos. Este XML contiene una estructura que asocia a una etiqueta determinada el significado correspondiente a cada uno de los idiomas. Por ejemplo, en el Algoritmo 14 se muestra el ejemplo más simple de XML aplicado al multilingüismo.

La ampliación de una página ya creada, resulta sencilla gracias a este formato. Se realiza dependiendo de como se quiera ampliar:

- Si la ampliación es un nuevo idioma, solamente hay que añadir entre dos nuevas etiquetas que lo diferencien, la traducción correspondiente.

Algorithm 14 Ejemplo de una página XML multilingüe.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<toctoc>
  //Etiqueta de párrafo.
  <hola_mundo>
    //Etiquetas de idioma junto con el significado.
    <esp>Hola mundo</esp>
    <eng>Hello Word</eng>
  </hola_mundo>
</toctoc>
```

- Si se quiere añadir un nuevo párrafo, habrá que crear una etiqueta de párrafo que lo diferencie. Dentro habrá que escribir las diferentes versiones en cada uno de los idiomas deseados.

Modificación de los archivos PHP. Los archivos de PHP se encargan de mostrar el contenido adecuado que se recoge en los archivos XML. Para ello, PHP tiene que hacer dos pasos:

- Abrir el archivo XML adecuado y obtener el texto. Esto de hace mediante:

  include("include/XML.inc");
  $fichero="idiomas/ejemplo.xml";
  abreXML($fichero,$arr_vals, $arr_index);

  La primera línea incluye las funciones que se usan más tarde. La variable *$fichero* contiene la ruta al XML relacionado. La función *abreXML()* abre el fichero indicado y guarda el XML en forma de matriz dentro de la variable *$arr_vals* que será utilizada más tarde.

- Imprimir el párrafo adecuado en el idioma deseado. Esto se hace con la función:

  obtieneCampoXML($arr_vals, $PARRAFO, $IDIOMA, $SALIDA); echo $SALIDA;

  Esta función obtiene el párrafo etiquetado por la variable *$PARRAFO*, escogiendo el idioma *$IDIOMA* y lo guarda en la variable *$SALIDA*. Imprimir esta última variable mostrará el texto.

Estos pasos tendrán que repetirse para cada página PHP que contengan texto y por cada uno de los párrafos escritos.

Añadir un nuevo idioma. Para añadir un nuevo idioma, aparte de las traducciones pertinentes en cada página, hay que añadir una nueva bandera en la página principal. Cada bandera recarga la página enviando una variable que contiene el código del idioma, que es el que se utilizará a su vez en el XML para diferenciarlos.

B.5.4. Conexión a Internet.

La conexión a Internet viene definida dentro de la página *redWireless.inc* que contiene todas las funciones necesarias para su manejo. Primero se presenta una explicación de cada una de ellas:

- *escribe_log*(): guarda una cadena dentro del log del sistema.
- *reduce_log*(): deja en el log solamente aquellas entradas que tienen menos de un periodo de tiempo definido por la constante *TIEMPO_LOG*. El resto de entradas se guardan en la copia de seguridad del log.
- *da_internet_usuario*(): se encarga de generar las colas de TC específicas para el cliente que invoca a esta función. De esta manera se permite el acceso de este a Internet.
- *mantiene_internet_usuario*(): esta función se encarga de que el ordenador principal no dé de baja al cliente que la invoca. Debe ser llamada periódicamente o el sistema pensará que el cliente a perdido su conexión.
- *obtiene_estado_cliente*(): obtiene del sistema el estado del cliente, es decir, comprueba si el sistema da por conectado o desconectado este cliente.
- *obtiene_estado_sistema*(): indica si TocToc está encendido o apagado.
- *cierra_internet_usuario*(): cierra correctamente la conexión de un usuario.

De todas estas funciones la más importante es *da_internet_usuario*(). Esta es la función que hay que revisar si se pretende cambiar la funcionalidad de la conexión a Internet. Aquí habrá que indicar la nueva cola si se quiere usar algún otro tipo de cola o es donde se añadirá el IMQ en caso de que se quiera utilizar.

B.5.5. Generación de la ayuda del sistema.

La generación de la página de ayuda viene facilitada por dos funciones de PHP. Estas funciones se encargan de formar la estructura del índice de ayuda, los enlaces entre el índice y la explicación, y la propia explicación. Estas funciones:

- *ayudamenu($arr_vals, $ANCLA, $TEXTO)*: Esta función genera el menú de ayuda, mostrando el texto indicado en la página XML por la etiqueta incluida en la variable *$TEXTO* y que llevará al párrafo que tenga el enlace *$ANCLA*. La variable *$arr_vals* es, como se ha explicado anteriormente, la que contiene el texto obtenido de la página XML.
- *ayudaexplicacion($arr_vals, $TITULO, $EXPLICACION, $ANCLA)*: genera el párrafo explicativo al que se dirigirá el enlace del menú con el enlace *$ANCLA* y que se compondrá por un título y su explicación compuestos por el texto que devuelva el XML a pasarle las variables *$TITULO* y *$EXPLICACION* respectivamente.

Por tanto, para cada entrada en la ayuda habrá que incluir cada una de las funciones anteriores, generando así un índice principal y una explicación que aparecerá más abajo.

B.5.6. Ampliación general de las páginas PHP.

Por motivos de facilidad de desarrollo futuro, cada página que se utiliza en el sistema incluyen las páginas *cabecera.inc* y *pie.inc*. La primera de ellas se importa justo al principio del documento, mientras que la segunda se incluye justo al final. La finalidad de estas dos páginas es la aplicación de un elemento común en todas las páginas de forma sencilla. Por ejemplo, en *cabecera.inc* se incluyen todos los META-TAGs que se utilizan en todas las páginas, de forma que el mantenimiento de estos resulta muy sencillo en todas las páginas. El archivo *pie.inc* está en este momento vacío y se puede utilizar para añadir elementos al final de la página, como por ejemplo, contadores de visitas y otros elementos comunes como logotipos.

B.5.7. Archivos de protección.

El arquitectura TocToc incluye algunas funciones de protección para que no existan accesos no autorizados en el sistema. Estas funciones están en los archivos *proteccion.inc* y *proteccion_ internet.inc*. El primero de los dos archivos, contienen unas funciones que impiden el acceso a una página a aquellos navegadores que no hayan generado una sesión de PHP. Como la sesión solamente se genera si el usuario entra en el sistema introduciendo su nombre de usuario y su contraseña, solamente los usuarios dados de alta pueden tener acceso al sistema. El segundo archivo, *proteccion_ internet.inc*, es similar al primero salvo que impide el acceso a ciertas páginas a los usuarios que no tienen permitida la conexión a Internet.