ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y TELECOMUNICACIÓN

Titulación:

INGENIERÍA INFORMÁTICA

Título del proyecto:

CONTROL REMOTO DE LOS PRINCIPALES DISPOSITIVOS DE UN ORDENADOR MEDIANTE UN CLIENTE WEB.

Alumno: Javier Ladrón Arroyo

Tutor: José Javier Astrain Escola

Pamplona, 16 de Febrero de 2015

INDICE

1.1 DESCRIPCIÓN Y OBJETIVOS

Se trata de desarrollar un sistema que sea capaz de gestionar los diferentes dispositivos de un ordenador personal, por parte de un usuario final, de una manera fácil, intuitiva y atractiva. Se pretende "estandarizar" la monitorización de un determinado espacio sin que el usuario tenga que invertir ningún tipo de esfuerzo (económico, material...), de manera que con la tecnología que, generalmente, un usuario "convencional" posee en su día a día (ordenador personal, conexión a Internet...) pueda monitorizar un entorno por webcam, micrófono y control de un teclado.

El sistema debe proporcionar acceso a los dispositivos webcam, micrófono y teclado del ordenador que se pretende monitorizar. Se pretende que el usuario pueda gestionar tanto los datos recibidos generados por los distintos dispositivos (imágenes, grabaciones, ficheros) como modificar los parámetros de monitorización de los mismos. Además es absolutamente fundamental que el usuario pueda acceder al sistema desde cualquier parte, en cualquier momento. Para que el sistema gestor pueda estar accesible desde cualquier terminal de manera pública se implementará a través de un servidor web. Lo mismo exactamente se utilizará para gestionar y sincronizar los datos y resultados obtenidos por la webcam, micrófono y teclado del ordenador monitorizado.

El usuario final contará con dos interfaces para ver y gestionar el sistema. La primera de ellas será una página web diseñada específicamente para ordenares y tablets. La segunda de ellas es una aplicación para smartphones con sistema operativo Android para hacer uso del sistema de manera más clara y eficiente para este tipo de dispositivos.

La página web constará de una parte para cada dispositivo donde se podrán modificar los parámetros de monitorización, enviar una petición al sistema para el uso del dispositivo y una bandeja de entrada para ver los datos obtenidos. Además, para cada resultado obtenido se debe poder visualizar, compartir (con otros usuarios del sistema), enviar por email y borrar cada uno de ellos. Habrá también una zona donde el usuario podrá ver los datos compartidos por otros usuarios con él. Dispondrá de un registro de actividad para ver todas las actividades llevadas a cabo por el usuario en el sistema. Por último la web dispondrá de una zona para la edición de los datos personales de la cuenta, nombre de usuario, contraseña, email...

La aplicación Android es similar pero su funcionalidad es menor que la web, ya que está optimizada para smartphones. Básicamente contiene lo mismo que la web pero no incluye la zona de compartición de ficheros con otros usuarios, el registro de actividad y el panel de configuración de usuario.

Ambas interfaces son de carácter privado. El acceso se realizará mediante un usuario y contraseña.

Para la obtención del control de los dispositivos principales se diseñará y desarrollará un software local que se ajustará a los parámetros de monitorización editados por el usuario, obtención de datos de los diferentes dispositivos y la comunicación y sincronización de dichos datos con el servidor web. Una característica fundamental del software es que debe garantizar

la autenticidad del usuario (nombre usuario y contraseña) con el ordenador donde está en ejecución dicho software (uno y sólo un usuario es capaz mediante un nombre y contraseña, de poder monitorizar dicho ordenador).

Otra herramienta que se desea incluir en el proyecto es un software para poder obtener el control remoto de un ordenador a través de otro.

Por último, se dotará al sistema de un par de herramientas sobre inteligencia artificial. Se incorporará un mecanismo de reconocimiento de patrones sobre sintaxis del idioma español y un sistema de reconocimiento facial para un futuro posible mecanismo de autenticación.

Más adelante, en los apartados de "Solución al problema" en la introducción y en "Semántica del proyecto" en el diseño se especificarán mucho más detalladamente las características del proyecto.

1.2 Estado del arte

En este apartado voy a describir el conjunto de tecnologías que existen actualmente en el mercado y que son competencia directa con las que yo he utilizado para llevar a cabo el desarrollo del proyecto.

Para el software encargado de interactuar con los principales dispositivos del ordenador me he decantado por Microsoft .NET (C#). Por ello voy a hablar del principal competidor de .NET en cuanto a lenguajes para software de escritorio, Java. Para ello me he basado en un artículo de la Universidad Jaime I, Castellón de la Plana [1].

Java es un lenguaje multiplataforma, compatible con cualquiera de los sistemas operativos punteros del mercado. Fue diseñado para trabajar de manera segura en red y para sustituir de manera eficaz código nativo. La principal característica de este lenguaje es que es compilado e interpretado en una máquina virtual, en lugar de compilado sobre el kernel de un sistema operativo concreto. De esta manera es como consigue la independencia de plataformas.

Java es un lenguaje con un paradigma orientado a objetos. Pero con el paso de los años ha ido evolucionando para satisfacer a todos los entornos en los que puede ser utilizado (web, webservices, local, tecnología móvil...).

Como he dicho antes, Java está diseñado para mantener fuertes controles de seguridad. Tanto a nivel de programación de usuario como a nivel de máquina virtual. Todas las instancias de las clases son pasadas por referencia mientras que atributos primitivos son pasados por valor. Cuando un programa está en ejecución y llega a su fin es la propia máquina la que se encarga de liberar la memoria principal utilizada. Este es un ejemplo de seguridad a nivel de máquina virtual. Un ejemplo de seguridad a nivel de programación de usuario es la comprobación estricta de tipos durante tiempo de compilación. Asi garantiza que no se produzcan desbordamientos de pila. Otro ejemplo muy claro es el manejo de excepciones. Siempre que un programador utilice código que interactúe de alguna manera con el sistema (dispositivos entrada/salida del sistema, conexiones con base de datos, conexiones de red, valores nulos no válidos, operaciones de desbordamiento de pila...) está obligado a capturar posibles errores que puedan producirse en tiempo de ejecución y detengan de forma fatal la ejecución del programa.

Otras caracteríticas de Java son que posee un *gestor de seguridad* que evita el acceso a recursos del sistema, está preparado para programación concurrente sin necesidad de utilizar ninguna biblioteca externa.

Como final a esta explicación del lenguaje de programación Java comentar que existe cierta corriente de personas que argunmenta que Java es un lenguaje lento porque debe interpretar los bytecodes a código nativo antes de poder ejecutar un método, pero gracias a la tecnología JIT, este proceso se lleva a cabo una única vez, después el código nativo se almacena de tal modo que está disponible para la siguiente vez que se llame. De esta forma únicamente es compilado e intepretado la primera vez.

A continuación voy a exponer las principales características de MATLAB en lo referido a la captura y procesamiento de imágenes en contraposición a OpenCV y AForge que son las que voy utilizar en el desarrollo del proyecto para los objetivos previamente mencionados.
En este punto me voy a centrar más en los pros y contras de haber escogido OpenCV en lugar de Matlab. De esta manera quedarán contrastadas ambas tecnologías. Dicha información ha sido obtenida de la materia *Computer Vision*, impartida en la universidad Politécnida de Turín.

Matlab es un lenguaje muy amplio que posee librerías destinadas al procesamiento de imagen. OpenCV en cambio es justo una librería para el procesamiento optimizado de imagen. OpenCV es capaz de procesar 30 frames por segundo mientras que Matlab únicamente 5. Además Matlab necesita muchos más recursos del sistema, por lo que es menos eficiente que OpenCV. Por otra parte Matlab tiene su propio y exclusivo IDE mientras que OpenCV puede ser integrado en los IDEs más utilizados del mundo como Eclipse, Netbeans, Visual Studio, XCode... OpenCV tiene su versión original destinada para C++ pero posee traducciones y *wrappers* para Java, python, Rubí... Para concluir estas comparaciones hablaré del precio de las licencias. Matlab es una plataforma de pago mientras que OpenCV es una tecnología OpenSource.

El siguiente punto a evaluar es la teoría y tecnología utilizada para la realización de un procesador del lenguaje. En este proyecto es indispensable la realización de un párser para el reconocimiento de patrones sobre terrorismo en español tal y como se detallará más adelante. Para la especifiación de la gramática correspondiente me he decidio por usar la herramieta GOLD Parser Builder, que más adelante documetaré detalladamente. En este apartado, en cambio, analizaré las característica de herramientas competidoras muy conocidas como son flex y bison. Esta información, esta recogida de un website destinado a herramientas de compilación llamado *compilertools* [2].

Flex es una herramienta para generar analizadores léxicos (programas que reconocen palabras de un texto). Flex lee los datos de una determinada entrada, entrada estándar como la pantalla, fichero de texto... para ser analizada por una descripción de un escáner (analizador léxico) a generar. Dicha descripción consiste en parejas de expresiones regulares con fragementos de código a ejecutar escritos en C. Con una serie de ficheros y rutinas de la librería produce un ejecutable en el lenguaje de programación C que se ejecuta cada vez que se lee un token. Se puede definir un token como la unidad mínima dentro de un *universo* (un texto).

Bison, en cambio, es un generador de analizadores sintácticos (párser) de caracter general que, a partir de la descripción de una gramática LALR(1) libre de contexto y con los *tokens* que va recibiendo del analizador léxico previamente creado (con Flex) reconoce o no ciertos patrones indicados en las reglas de la gramática.

Ambas herramientas tiene licencia GNU por lo que son de código abierto.

La siguiente tecnología que voy a exponer es ASP.NET. Entra en contraposición con PHP, que es el lenguaje en el que desarrollaré toda la parte lógica y funcional del servidor web del sistema. Esta documentación ha sido obtenida de la página web oficial de Microsoft [3].

ASP.NET es una tecnología, desarrollada por Microsoft, para el desarrollo web unificado que incluye los servicios necesarios para la creación de websites totalmente profesionales con mínimo código.

ASP forma parte del framework .NET y al codificar las aplicaciones ASP tiene acceso a las clases del framework .NET. El código de las aplicaciones puede ser implementado en cualquiera de los lenguajes compatibles con el *Common Language Runtime* (CLR), entre ellos Visual Basic, C#, J#, JScript .NET. Todos estos lenguajes son compatibles con ASP y se benefician de las características del CLR como seguridad de tipos, herencia...

ASP incluye:

- marco de trabajo de página y controles.

- compilador de ASP

- infraestructura de seguridad.

- funciones de administración de estado

- configuración de aplicación.

- supervisión de estado y características de rendimiento.

- capacidad de depuración

- marco de trabajo de servicios web XML

-entorno de host extensible

-administración del ciclo de vidad de las aplicaciones

- entorno de diseñador extensible.

Me gustaría acabar esta exposición de ASP.NET con una explicación un poco más detallada del compilador de ASP.

Se compila todo el código de ASP.NET, lo que permite el establecimiento inflexible de tipos, las optimizaciones de rendimiento y el enlace en tiempo de compilación, entre otras ventajas. Una vez que se ha compilado el código, el Common Language Runtime compila una vez más código de ASP.NET en código nativo, lo que permite un mayor rendimiento.

ASP.NET incluye un compilador que compilará todos los componentes de la aplicación, incluidas las páginas y los controles, en un ensamblado que el entorno de host de ASP.NET puede utilizar a continuación para atender las solicitudes del usuario.

Por último, en este apartado de *Estado del arte* voy a exponer las principales características de Objective C, el lenguaje utilizado para desarrollar toda la programación lógica y funcional de las aplicaciones para dispositivos móviles con sistema operativo iOS de Apple. Esta documentación ha sido obtenida de la web oficial de Apple Inc [4].

Objective C es un lenguaje de programación primario usado para desarrollar sofware para los sistemas operativos de Apple, Mac OS X para ordenadores y iOS para tablets, smartphones y reproducteres de música.

Objective C extiende al lenguaje de programación C incorporando el paradigma orientado a objetos (creación de clases, herencia...) y tiempo de ejecución dinámico en contraposición a C que es un lenguaje con un paradigma de programación estructurado. De C hereda la sintáxis, los tipos primitivos, el flujo de las sentencias de control (if,else, while...0). Como he comentado previamente añade sintáxis para la creación de clases y métodos.

Por otra parte añade soporte de idiomas para la gestión gráfica de objetos y objetos literales.

La característica más relevante es que permite ciertas tomas de decisiones en temas de tipado... en tiempo de ejecución en lugar de en tiempo de comilación.

Por último, concluir que al ser un lenguaje cerrado para sistemas Apple debe ser desarrollado en el IDE de Apple XCode.

1.3 Solución que propongo

De acuerdo a la descripción y problemas planteados del proyecto voy a proceder a explicar cómo he abordado todas y cada una de sus partes.

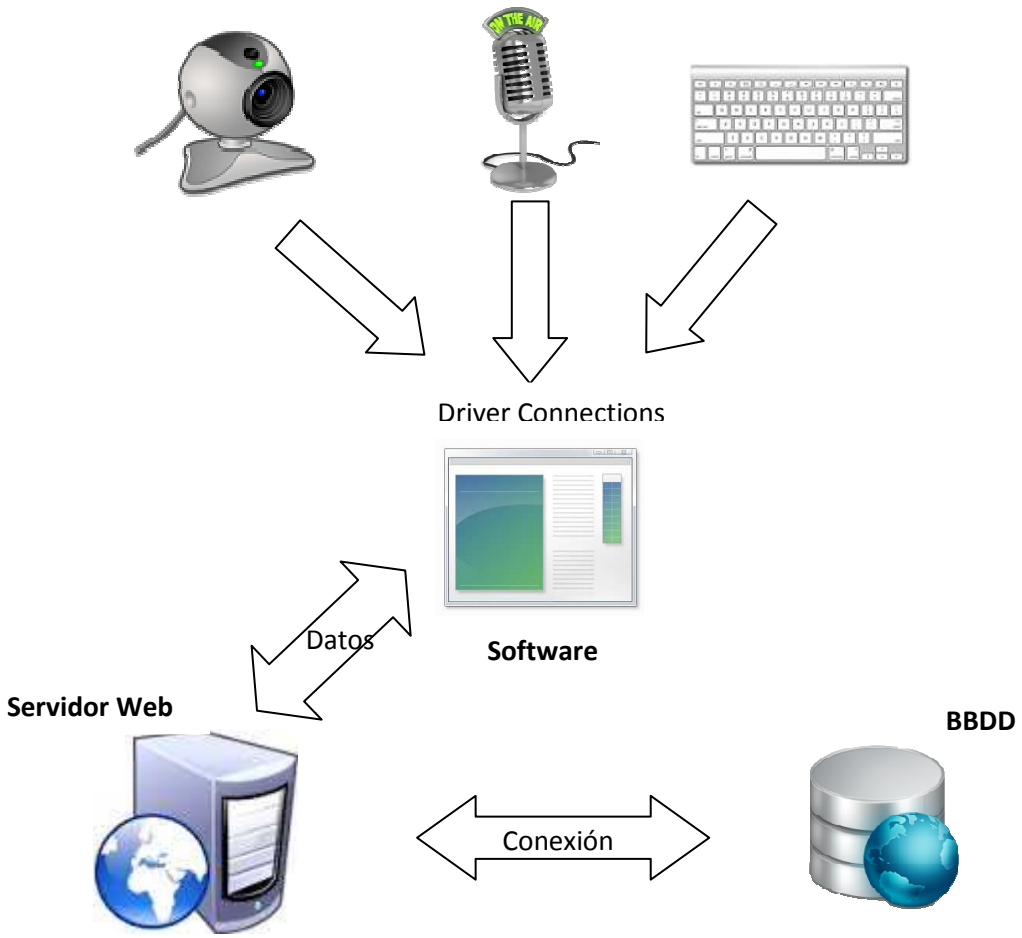
Sobre el software encargado de interactuar con los principales dispositivos del ordenador he optado por desarrollarlo en el lenguaje C# con la librería gráfica de Microsoft, .NET.

Este programa deberá importar librerías generales para la comunicación con el micrófono, webcam y teclado del ordenador en el que se está ejecutando. Dicha elección supone que el servicio, en un principio sólo está disponible para ordenadores con sistema operativo Windows.

Es OBLIGATORIO que el programa responda a una, y sólo a una, autentificación (correspondiente al usuario del ordenador) y NECESARIO que el software sea multiproceso de manera que pueda hacer comunicarse de manera concurrente con los dispositivos principales, comprobar posibles peticiones del usuario y coordinar el protocolo de transmisión de ficheros y datos con el servidor web.

Además, se desarrollará dentro del software un módulo de registro para el usuario, ya que será el único mecanismo habilitado para la incorporación al sistema.
Asi mismo, el usuario que quiera darse de baja en el sistema también lo tendrá que hacer desde este módulo.

Ahora voy a mostrar un esquema de cómo debería funcionar el software.



En cuanto a la herramienta de gestión del sistema por parte del usuario propongo dos posibles soluciones.

La primera de ellas es una página web. Destinada a terminales con una resolución de pantalla superior (ordenadores de mesa, portátiles, tablets...).

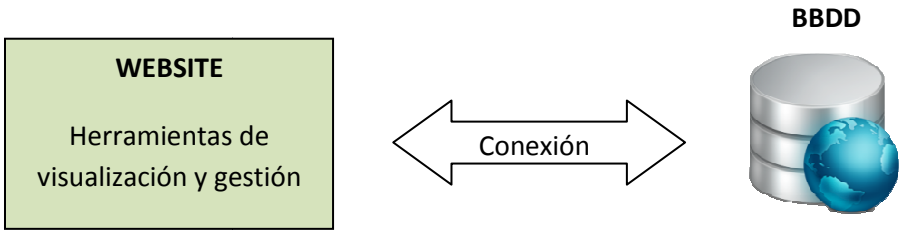
Dicha web estará accesible para cualquier usuario registrado. Se deberá acceder con usuario y contraseña para ver su contenido (propio del usuario).

Se mostrará en secciones claramente diferenciadas cada uno de los contenidos y acciones que se podrán visualizar y llevar a cabo de cada uno de los periféricos. Se pretende que dentro de la web se obtenga un registro de todas aquellas acciones que el usuario va llevando a cabo en el sistema.

Además, como se pretende dar ciertas funcionalidades de red social entre usuarios del sistema a la hora de compartir contenidos, la web dispondrá de una zona en la que se podrá ver dichos contenidos.
Obviamente el usuario debe poder modificar sus datos personales: nombre, apellidos, usuario, password, email... Se habilitará una zona donde el usuario pueda llevar a cabo todas estas operaciones.

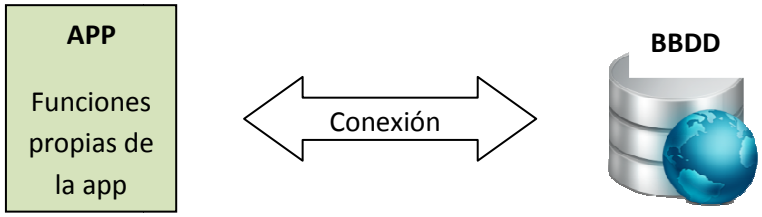
Por último, la web dispondrá de una zona donde el usuario podrá descargar todos los softwares de los que dispone el sistema. A saber, el software controlador de los dispositivos, el software para el control remoto y la aplicación Android del sistema.

Esquema de la página web:



La segunda de ellas es una aplicación Android. Obviamente está destinada a terminales con menor ratio de interfaz gráfica (smartphones con el sistema operativo Android, de Google). Básicamente tendrá que disponer de casi todos los elementos pensados para la página web. Incorporará autentificación por parte del usuario y los paneles de contenidos y acciones de los dispositivos. En principio no considero desarrollar el resto de funciones previamente comentadas, ya que el objetivo de esta segunda herramienta de gestión es hacerla rápida y con la funcionalidad imprescindible para el tipo de terminales donde será compatible.

Ambas interfaces (web y app) estarán absolutamente sincronizadas. El usuario siempre podrá utilizar ambas.
Esquema de la app:



Por último voy a proponer la solución que en principio tengo pensada para abordar el software complementario de control remoto. Este software tendrá que disponer de dos partes claramente diferenciadas.

La primera de ellas actuará como servidor y será el encargado de ofrecer el control total del terminal donde se está ejecutando. Recibirá órdenes a través de una conexión siempre abierta, las interpretará y devolverá unos resultados. Su principal función será, como digo, ejecutar las instrucciones que le lleguen por la conexión y enviar por la conexión una representación de la salida estándar (pantalla). Se pretende llevar a cabo un sistema de autentificación mediante reconocimiento facial.

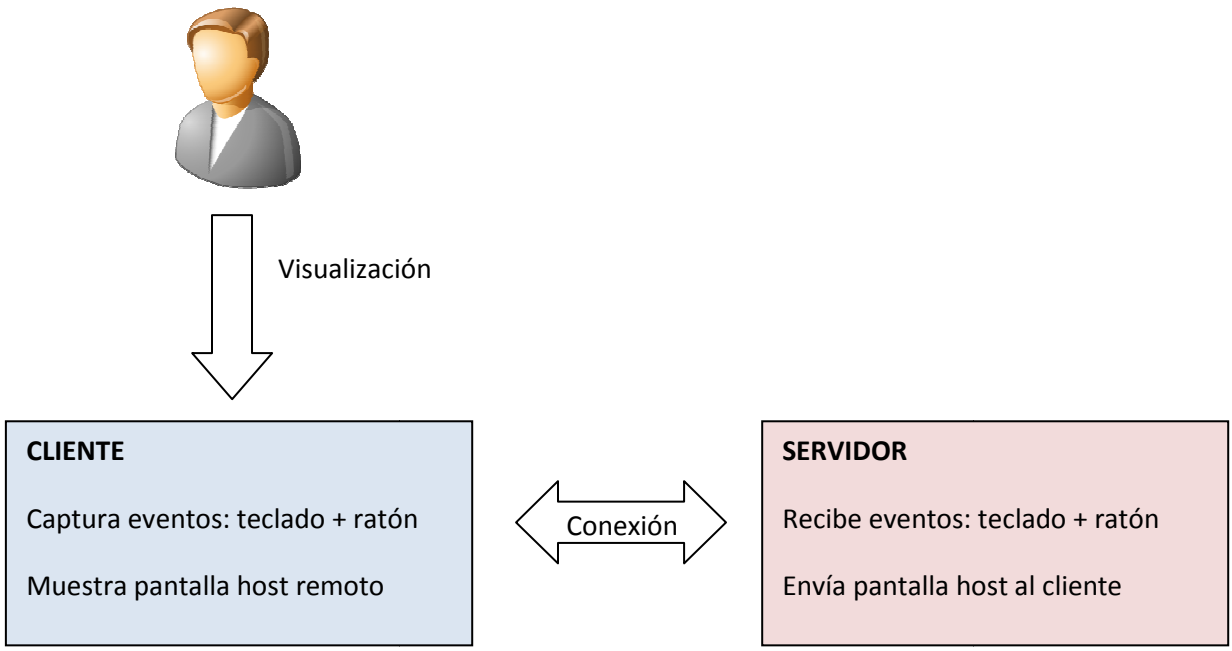
La idea inicial es desarrollarlo en el lenguaje multiplataforma Java precisamente para aprovechar el hecho de su compatibilidad con todo sistema operativo, Windows, Mac OX, distribuciones GNU Linux...

La segunda de ellas actuará como cliente y proporcionará al usuario una interfaz gráfica del ordenador que está monitorizando. El cliente mantendrá una conexión abierta con el servidor. Este programa capturará los eventos de teclado y ratón del ordenador donde se está ejecutando y los enviará al servidor. Entonces recogerá la representación visual de la conexión enviada por el servidor y se la mostrará al usuario.

Además de la funcionalidad principal, se pretende dotar de alguna otra funcionalidad extra como el intercambio de ficheros, ejecución de comandos de la webcam, calidad de servicio de la funcionalidad principal...

Al igual que el servidor, el lenguaje en el que se desarrollará esta segunda parte del software será en Java para aprovechar las ventajas que he comentando en el párrafo anterior.

Esquema del funcionamiento del software:



2.1 Semántica

En este apartado procederé a la explicación más minuciosa y detallada del proyecto.

Se pretende desarrollar un sistema que permita a un usuario controlar los principales dispositivos (webcam, micrófono y teclado) de un terminal (ordenador) de manera facil, intuitiva y sencilla.

Para ello de cada usuario se requiere saber un identificador único que actuará como clave primaria, nombre, apellidos, usuario, password, email y un código absolutamente diferencial para garantizar la exclusividad del ordenador que quiere monitorizar. Dicho código será la dirección MAC de dicho ordenador.

En el apartado de dispositivos, de cada webcam se requiere conocer un identificador único (clave primaria), el tiempo de monitorización que se le indica a un software controlador (será explicado más en detalle en este apartado) con el que se conoce cada cuanto tiempo dicho software debe comprobar si existe o no una petición para la webcam por parte del usuario, el número de imágenes a capturar en cada petición, el intervalo de tiempo entre cada una de ellas y un parámetro que indique si dicha webcam ha recibido una petición o no durante el periodo del tiempo de monitorización. Obviamente un usuario puede controlar una única cámara (o ninguna si el terminal no posee) y cada cámara es controlada por un único usuario (identificador del usuario como clave foránea en la webcam).

Para un micrófono ser quiere conocer un identificador único (clave primaria), tiempo de monitorización, tiempo (en segundos) que el software controlador debe grabar en cada petición y el parámetro que indica si el micrófono ha recibido o no una petición del usuario durante el tiempo de monitorización. La relación con el usuario es exactamente la misma que con la webcam.

Con el último de los dispositivos que se pretende monitorizar, el teclado, se requiere lo mismo que con el micrófono. Un identificador único (clave primaria), tiempo de monitorización, tiempo (en segundos) de captura de pulsaciones por parte del software controlador y el parámetro que indica si se ha recibido o no una petición por parte del usuario. La relación con el usuario es la misma que la de la webcam y el micrófono.

Cada uno de los dispositivos especificados puede generar un determinado tipo de archivo. La webcam podrá tomar imágenes (fotografías) en formato JPG. De cada foto es necesario conocer un identificador único (clave primaria), un nombre formado por la fecha y hora de la captura, un campo donde se refleja el path del servidor donde estará alojada físicamente la foto y la fecha en la que se subió. Un usuario podrá tomar las fotos que desee y cada foto será exclusiva de cada usuario (Id usuario como clave foránea en la foto).

En cuanto al micrófono, decir que generará ficheros de audio WAV. De cada grabación se requiere un identificador (clave primaria), un nombre (con a fecha y hora) , el path y la fecha. Un usuario podrá realizar todas las grabaciones que desee y cada grabación será propia de un único usuario.

Por último la monitorización del teclado permitirá al usuario salvar ficheros PDF con las teclas pulsadas durante el tiempo de captura de una petición. De cada fichero PDF se requiere un identificador (clave primaria), un nombre (con a fecha y hora) , el path y la fecha. Un usuario podrá realizar todas las capturas de teclado que desee y cada captura de teclado será propia de un único usuario.

Es imprescindible, para el llevar a cabo un cierto control del sistema por parte del usuario, que toda actividad generada por éste, quede registrada en el sistema. De cada acción (inserciones, borrados...) es necesario saber un identificador único de la actividad, que actuará como clave primaria, una descripción en el que se detalla el contenido de la actividad, fecha, hora y tipo de actividad que es. En principio los tipos de actividad que quedarán registradas serán de inserciones de archivos por parte de los diferentes dispositivos, borrado de archivos y diferentes alarmas generadas por el sistema que se detallarán más adelante. Cada actividad es propia de cada usuario.

Por otra parte, el sistema pretende permitir la opción de compartir archivos con otros usuarios del sistema. Para cada archivo compartido en necesario un identificador (clave primaria), nombre del archivo compartido, el path donde está alojado en el servidor, la fecha, el tipo de archivo que es (foto, grabación o fichero PDF) y el identificador del usuario con el que ha sido compartido. Un usuario puede compartir los archivos que desee las veces que desee. Pero cada archivo compartido actuará de manera única siendo compartido por un único usuario.

Una vez declarada y especificada la estructura de datos que tendrá el sistema voy a proceder con la especificación de cada uno de los componentes software de los que se compondrá el sistema.

El primero a especificar será el software controlador. Su función principal es conectar con los dispositivos de monitorización del sistema, webcam, micrófono y teclado, capturar sus archivos y comunicarse con el servidor web.

Pero vayamos por partes. Lo primero que ofrece dicho software es un modulo de registro. En dicho módulo se le pide al usuario su nombre, apellidos, nombre de usuario y contraseña (dos veces). Este modulo comprueba las prestaciones de los campos (contraseñas que coincidan, nombre de usuario no cogido...) y procede al registro del usuario. Para el registro del usuario se llevan a cabo tres acciones:

1) Averigua la dirección MAC de la tarjeta de red del ordenador (más adelante explicaré el motivo).

2) Inserta en la base de datos un nuevo usuario con los datos pedidos más la dirección MAC del ordenador.

3) Inserta una nueva webcam, micrófono y teclado en la base de datos con unos parámetros por defecto.

Una vez hecho eso comunica en una salida estándar del software si se ha llevado a cabo o no correctamente el registro del usuario.

A continuación el usuario ya puede autenticarse en el software. Para ello introduce su nombre de usuario y contraseña. El sistema comprueba sus credenciales junto con la dirección MAC con la que se registró el usuario. De esta manera el sistema garantiza siempre que el ordenador donde está ejecutándose el software sólo el usuario que se registró puede autenticarse en él. Resulta obvio decir que sólo un usuario puede registrarse en el sistema con el software en un mismo ordenador (la dirección MAC, que es única de cada tarjeta de red está asociada a uno y sólo a un usuario).

Una vez autenticado correctamente en el sistema el software empieza a monitorizar de manera concurrente las posibles peticiones de cada uno de los tres dispositivos. Tanto los parámetros de monitorización como los parámetros propios de cada dispositivo (enumerados de manera específica antes) pueden ser modificables desde este software una vez que el usuario ha sido autentificado correctamente.

El software comprueba que una nueva petición ha sido solicitada de alguno de los dispositivos lleva a cabo la acción correspondiente de manera concurrente al proceso principal del software (por ejemplo: 3 fotos con un intervalo de 7 segundos o una grabación durante 18 segundos...). Los archivos resultantes se guardan en un directorio temporal y otro proceso concurrente es el que se encarga de subirlos al servidor. Concurrentemente, el sistema debe seguir monitorizando los tres dispositivos. Una vez subido los archivos el sistema notifica al servidor la actividad que se ha llevado a cabo según el tipo que sea.

El proceso encargado de la captura del teclado (keylogger) tiene a su vez una funcionalidad extra además de la ya comentada en el párrafo anterior. Una vez que el proceso ha capturado todas las pulsaciones durante el tiempo de captura, antes de guardarlo temporalmente en un fichero PDF, debe comprobar si el texto capturado se ajusta o no a una gramática implementada sobre un parser en el software sobre terrorismo. Dicho parser tiene que estar totalmente integrado en el software (forma parte de él). Si resulta que el texto capturado se ajusta a la gramática del parser el software debe generar una alarma que consiste en dos partes. La primera de ellas es notificar al servidor una actividad sobre la alarma. La segunda de ellas consiste en notificar al servidor que debe enviar al usuario un email de "emergencia" notificándole que se ha producido una alarma.

El sistema en principio, sigue monitorizando indefinidamente las posibles peticiones del usuario sobre cada uno de los dispositivos. Pero el usuario siempre podrá desconectar el servicio desde el módulo de autenticación. En ese momento el software detiene todos los proceso activos y finaliza toda actividad. Para volver a ejecutar la monitorización del sistema el usuario debe autenticarse correctamente de nuevo.

Por último el software, una vez que el usuario se ha autenticado correctamente en el software el usuario podrá darse de baja en el sistema (única forma, al igual que el registro). Se le preguntará si está seguro de la acción que quiere llevar a cabo. Si responde que sí el sistema se lo notifica al servidor y tiene que eliminar al usuario de la base de datos y en cascada todo los que tenga que ver con él: dispositivos, fotos, grabaciones, ficheros, archivos compartidos, actividades... A partir de ese momento el software vuelve a estar disponible para el registro de un nuevo usuario en el ordenador que esta ejecutándose.

De la página web, encargada de proporcionar todas las herramientas al usuario para gestionar los recursos del sistema hablaré a continuación. La homepage pedirá al usuario el usuario y contraseña. Estos credenciales, en un principio, son los mismos que el usuario estableció cuando se registró con el software controlador en el sistema.

Si la autenticación es correcta el sistema accede al perfil privado (de cada usuario) de la página. Estará formado por un ménu superior con los siguientes items: una zona para la webcam, otra para el micrófono, otra para el keylogger, otra para los archivos compartidos, una para el registro de actividad y otra para el panel de cofiguración de datos relevantes del usuario en el sistema.

Para la zona de la webcam existirá un segundo submenú en el que se reflejarán la bandeja de entrada, una zona para una nueva petición de la webcam al software contralador y una zona de configuración de parámetros de la webcam. Si el usuario, en este segundo submenú accede a la bandeja de entrada el sistema deberá reflejar una vista en el que se muestre por organizado por días y orden descendiente una tabla con el numéro de imagenes capturadas por el sistema (cada fila contiene el número de imágenes de ese día), la fecha (aaaa-mm-dd) y tres botones de acciones generales sobre cada una de las filas de la tabla. El

primero de ellos, el botón “Enter” que una vez pulsado por el usuario entrará en una vista en la que el usuario podrá visualizar y gestionar individualmente cada una de las fotos de ese día. Esta vista la explicaré más detalladamente más adelante. Un segundo botón será el de “show in gallery” en el que la web mostrará una galería dinámica superpuesta con las imágenes de ese día. La galería mostrará en orden de captura las fotos. El usuario podrá ir pasandolas una a una (hacia delante o hacia atrás) o podrá pasarlas automáticamente con un autoplay. Existirá un botón de cerrar para cerrar correctamente dicha galería. Por último, mencionar que existira un elemento de borrado para cada fila (imágenes de cada día) con el que el usuario podrá borrar directamente todas las imágenes de dicho día. El sistema deberá borrar físicamente del servidor las imágenes de ese día y borrar de la base de datos todas las entradas relacionadas con estas imágenes. Deberá garantizar (como toda operación con la base de datos) las transacciones de manera correcta.

De la vista correspondiente a la llamada del botón “Enter” recién comentado, el sistema deberá mostrar indiviualmente cada una de las imágenes en una tabla. Cada elemento de la tabla constará de una mini representación de la imagen, el nombre de la imagen (generado automáticamente por el software controlador) y una serie de botones.

Dichos botones serán uno para visualizar individualmente en el formato de galería (antes explicado) la imagen, otro para enviar por correo dicha imágen, otro para compartir con otro usuario del sistema y otro para borrar esa imagen concreta del servidor y de la base de datos. Si el usuario elige la opción de enviar por correo la web enviará al usuario a una pantalla en el que podrá introducir directamente una dirección válida (el sistema lo comprueba) o una palabra clave relacionada con otro usuario del sistema en el que si existen coincidencias mostrará los emails correspondientes. En cualquier caso una vez elegido, por cualquier medio, una dirección válida de correo el sistema enviará un email a dicha dirección con la imagen adjuntado (attachment) y un contenido de correo con una cabecera del sistema generada automáticamente y el contenido de un text-field que se proporcionará al usuario en dicha pantalla antes de enviar por si el usuario quiere dejar algun mensaje al destinatario con la imagen adjuntada. En cambio, si el usuario elige la opción de compartir el sistema le enviará a una pantalla en el que el usuario eligira a un usuario del sistema con el que desea compartir dicha imagen. Una vez elegido el destinatario el sistema debe traspasar dicha imagen a la zona de ficheros compartidos, que explicaré más adelante del perfil del destinatario.

Hasta aquí he explicado la zona de la webcam en la web. Para el micrófono y el keylogger decir que muchas de las cosas cometadas serán exactamente las mismas que para la webcam. Por eso comentaré aquellas cosas que son diferentes a ella.

Tanto para micrófono como keylogger, en la bandeja de entrada, cada fila mostrara el número de grabaciones/ficheros pdf, fecha en el que fueron creados y las opciones de “Enter” y borrado general de dicho dichos archivos. El borrado general funciona exactamente igual para ambos que como en la webcam. El “Enter” lleva a una pantalla, como en la webcam en el que se muestra por filas, individualmente el nombre de la grabación/fichero pdf, una opción de descarga para el fichero pdf (descarga en el ordenador dicho fichero) o una opción de reproducción en el propio navegador en el caso de la grabación. Obviamente para ambos archivos (grabacion y fichero pdf) existe la opción de envío por correo,compartir y borrar individualmente que funcionará exactemente igual a los procesos explicados para la webcam.

En el menú principal existirá una zona donde el usuario podrá visualizar los archivos (fotos, grabaciones y ficheros pdf) que el resto de usuarios le han compartido. Los tres tipos se reflejarán en la misma vista. Se mostrar el nombre del archivo, el usuario que lo ha compartido

y una opción de borrado de cada archivo. Obviamente el archivo se podrá visualizar, reproducir o descargar dependiendo de su naturaleza (imagen, grabación o fichero pdf).

Aparecerá también una zona en el menú principal donde el usuario verá reflejado toda la actividad generada en el sistema debido a sus acciones. Todo el tema de subidas de archivos por parte del software controlador, borrado de dichos archivos, alarmas generadas … Concretamente se especificará de cada notificación el icono correspondiente según el tipo, una mini descripción de dicha notificación, la fecha y la hora en la que se produjo. Obviamente irán ordenadas en orden desdendiente por fecha y hora. Tendrá el usuario también un botón si quiere limpiar el historial de notificaciones de la actividad si considera oportuno eliminarlas pasado un tiempo.

Por último, el usuario tendrá en el menu principal un item de configuración en el que podrá modificar sus datos personales como usuario del sistema. Estos datos serán el nombre, apellidos, nombre de usuario, contraseña y correo eléctrónico. Solamente tener en cuenta que en el caso de que el usuario quiera modificar la contraseña deberá introducir en un campo la contraseña antigua e introducir doblemente en otros dos campos la nueva contraseña. Debe evaluarse a verdadero tanto la antigua contraseña como que la nueva coincidad en los dos campos. Del resto de datos solo debe comprobar que el email tenga un formato correcto.

He dejado como último gran apartado de esta especificación detallada el software de control remoto de una máquina desde otra. El sofware está dividio en dos grandes partes: un programa que actúa como servidor (máquina que será controlada) y un programa que actúa como cliente (puede controlar la máquina remota). En un primer lugar definiré el servidor y a continuación el cliente.

Para el servidor se pretende desarrollar un programa que a través de una conexión (puerto 8000) capture los eventos de teclado y ratón que le lleguen y los ejecute en la máquina que se está ejecutando. A su vez el programa debe realizar periódicas capturas de pantalla y enviarlas por la conexión para la otra parte del sofware (cliente). El programa debe ser multihilo. Es decir puede aceptar una o varias conexiones de un cliente. Y como he dicho antes, cada proceso debe ser capaz de sincronizar la captura y ejecucion de eventos (ratón y teclado) provenientes de la conexión abierta y el envío de las capturas de pantalla por la conexión periódicamente.

Una funcionalidad extra es la de la captura de una foto a traves de la webcam. Para ello el programa recibe por la conexión una “orden” distinta a a los eventos de ratón y teclado. Si se da el caso, el software realiza (de manera concurrente) una fotografía con la webcam y una vez obtenida la manda por la conexión abierta.

Como complemento el servido ofrece una última funcionalidad extra que debe ser la de escuchar por otra conexión (puerto 8001) distinta para ofrecer el servicio de un protocolo de transmisión de ficheros. Por dicha conexión envía o recibe el fichero solicitado por el cliente. Este servicio debe ser totalmente concurrente con la función principal recientemente explicada.

El software servidor posee una interfaz grafica con un botón con el que puede abrir y cerrar todas las conexiones cada vez que desee. De este modo controla cuando quiere o no ofrecer el servicio de control remoto. Antes de acabar el software servidor de control remoto es de recibo decir que debe disponer de un sistema de autentificación para que solo un usuario registrado pueda permitir el acceso a su equipo. Debe ser un sistema de

autentificación mediante reconocimiento facial. Para ello se desarrollará un prototipo de reconocimiento facial mediante el algoritmo de análisis del las componentes principales (PCA). El sistema estará basado en un entrenamiento con el que se extrerán los eigenvectores y eigenvalues del usuario. A partir de ese entrenamiento el prototipo (con una tolerancia más o menos permitida) debe permitir o restringir el paso al sofware servidor dependiendo si reconoce o no al usuario.

El software será desarrollado en Java para ser compatible con los principales sistemas operativos del mercado. La librería de visión por computador que se usará para el reconocimiento facial y la captura de fotos será opencv.

En cuanto al software cliente decir que es el que permite al usuario controlar el host remoto. Este software posee una interfaz gráfica con un menu y un "pantalla" donde se verán reflejadas las capturas de pantalla provenientes del servidor.
En primer lugar, dentro del menú, poseerá herramientas para la establecer la conexión (mediante la insercción de una dirección IP), cerrar la conexión, cerrar el cliente y cerrar el cliente y servidor. La función principal del cliente, una vez establecida la conexión (puerto 8000), es capturar los eventos de teclado y ratón producidos por el usuario y enviarlos por la conexión para que sean ejecutados en el servidor. A su vez, de manera concurrente, debe recoger las capturas de pantalla que le envia el servidor y reflejarlas en la "pantalla" de la interfaz gráfica para que el usuario pueda ver la monitorización del host controlado que está realizando. El software cliente además posee una serie herramientas adiciones. Las dos primeras es para descargar o subir un fichero en del/al servidor remoto. En ambos casos se debe escoger el fichero concreto junto a la ruta destino donde alojarlo. La transferencia de dicho fichero se realiza concurrentemente a través de otra conexión (puerto 8001). Otra herramienta es la de decir al sistema remoto el tomar una fotografía. Se le envia la orden por al conexión y espera a recibir por la conexión principal la imagen. Una vez recibida el cliente debe mostrar el usuario un cuadro de dialogo donde alojar la foto recibida.
También posee una herramienta para hacer el servidor invisible. Otra funcionalidad, bastante importante, que ofrecerá al usuario es el poder elegir la calidad del servicion en cuanto a las capturas de pantalla. Si el usuario detecta que la conexión ofrecida por la red está demasiado congestionada (se pierden capturas de pantalla) podrá bajar la calidad del servicio de manera que el número de bytes de catpura desminuya aún perdiendo calidad. En cambio bajará notablemente la congestión de la red haciendo el servicio mucho más fluido. Los parámetros que podrá elegir serán: 100% (máxima resolución), 80%, 66% y 50%. Por último, como herramientas adicionales, una vez que se ha cerrado la conexión el cliente proporciona al usuario la posibilidad de generar un informe con los datos de la conexión. Dicho informe deberá reflejar las direcciones ips de los hosts cliente y servidor, tiempo que ha estado mantenida la conexión entre ambos, tasa de la velocidad media de transferencia entre cliente y servidor y estadísticas apoyada también con un gráfico de sectores en la que quede perfectamente claro el número de capturas enviadas del servidor al cliente ya hayan sido correctamente recibidas o en cambio no han sido recibidas (fallo de conexión, fallo de red, problemas de congestión de red…).

2.2 Especificación de requisitos

En esta parte del análisis voy a centrarme en la especificación de requisitos.

Requisitos funcionales.
Como el sistema cuenta con cuatro partes bien diferenciadas enumeraré cada uno de los requisitos de forma diferenciada.

Software controlador

- Mecanismo para la creación de una y solo una cuenta asociada en un determinado terminal.

- Autentificación mediante usuario y contraseña para la cuenta que ha sido creada en dicho terminal.

- Monitorización de las posibles peticiones de uso de los tres dispositivos de manera concurrente.

- Modificación de parámetros de monitorización.

- Eliminación de la cuenta creada por el usuario.

- Implementación de un analizador sintáctico para el reconocimiento de patrones de texto sobre *terrorismo* en idioma español.

- Mecanismo para subir los ficheros generados (resultados) a un servidor web.

Página web

- Autentificación mediante usuario y contraseña.

- Para cada uno de los dispositivos (webcam, micrófono y teclado) debe constar un apartado con un menú con las siguientes opciones:

    1) Bandeja de entrada: donde el usuario puede ver los resultados generados (imágenes, grabaciones y ficheros) por el software controlador.
    2) Zona para solicitar al software controlador una nueva petición
    3) Zona donde se puede cambiar los parámetros de monitorización del dispositivo en cuestión.

- Para cada uno de los recursos obtenidos en la bandeja de entrada se deben poder acceder a ellos online, compartir con otros usuarios, enviar por correo y eliminar.

- Panel de visualización de recursos compartidos por otros usuarios del sistema. Se de poder acceder a ellos y eliminarlos.

- Zona donde se visualice el registro de actividad del usuario en el sistema. Se debe poder eliminar el historial si así lo desea el usuario.

- Zona de configuración para la modificación de datos personales del usuario.

- Apartado fuera de sesión donde el usuario podrá descargar todos los software disponibles del sistema (Software Controlador, Android App Gestor, Cliente Programa Control Remoto, Servidor Programa Control Remoto).

Aplicación Android

- Mecanismo de autentificación mediante usuario y contraseña. Si es autentificado correctamente las posteriores aperturas de la app llevaran correctamente al sistema sin que tenga que autentificarse de nuevo. Hará más fluida la app.

- Al igual que para la web dispondrá de un zona para cada dispositivo con donde existe un menú donde están las mismas funcionalidades que para la web.

- Logout de la sesión de la app. En este caso la próxima vez que se acceda si debe volver a pedir user y pass.

Software Control Remoto

Para el cliente:

- Establecimiento de conexión con el otro terminal.

- Cierre de conexión, cierre de conexión y server.

- Visualización completa de la pantalla del otro terminal. Correcta recogida y envío de eventos de teclado y ratón.

- Transferencia full-duplex de ficheros, gestión de calidad de servicio, servidor invisible.

- Tomar foto del otro terminal y poder acceder a ella.

- Generación de un informe de conexión.

Para el servidor:

- Autenticación mediante reconocimiento facial.

- Recogida y ejecución de eventos de teclado y ratón, sí como del resto de órdenes.

- Captura y envión de screenshots por la conexión.

Requisitos no funcionales

Software Controlador:

- Display en un interfaz gráfico para que el usuario pueda ver todas las acciones, errores...

- Mini-tutorial de cada dispositivo.

Página Web

- Menú superior horizontal con las principales secciones. Sub-menus verticales en las secciones de los dispositivos.

- Agrupamiento en forma descendente de los recursos en la bandeja de entrada por días. Eliminación de los recursos de cada día de forma grupal.

- Se verán los recursos que le han sido compartido al usuario de los tres dispositivos en la misma pantalla.

Android App

- Interfaz similar a la página web. Garantiza la consonancia de las herramientas cliente.

Software Control Remoto

Para cliente:

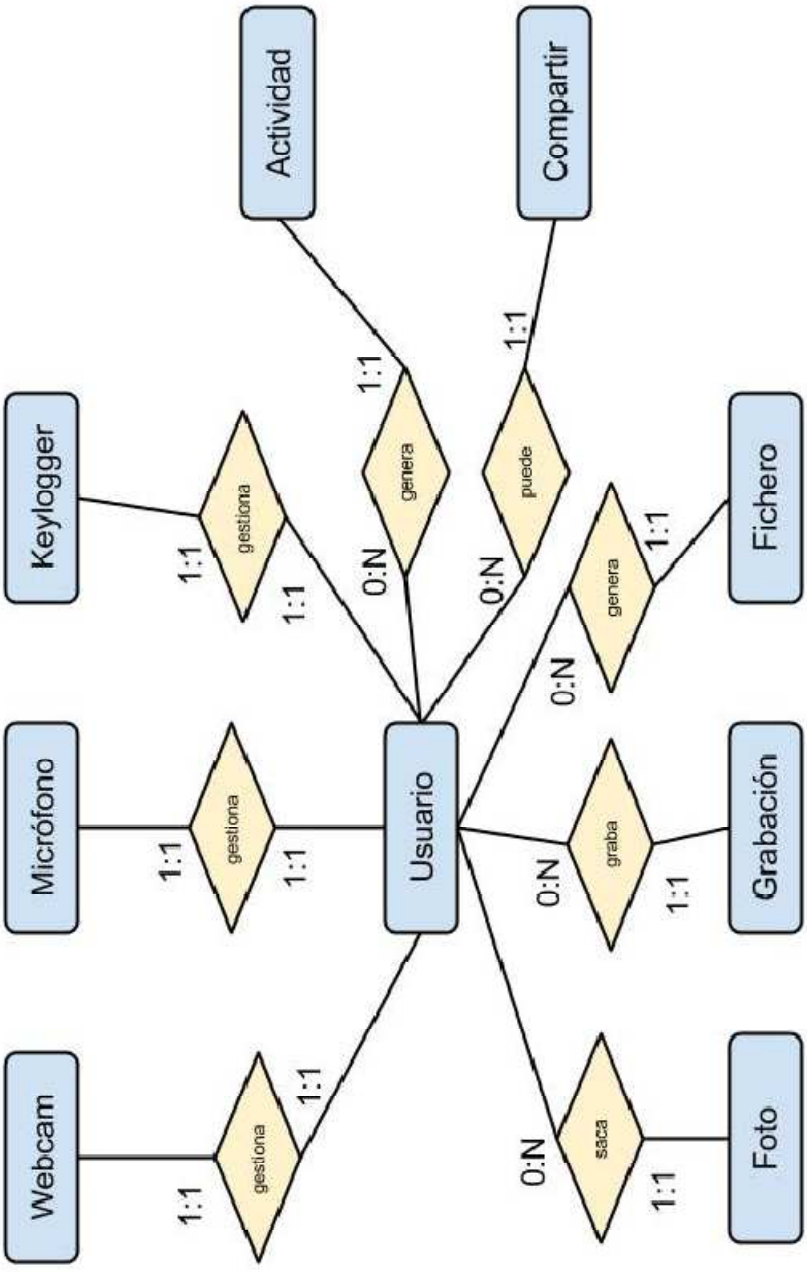
- Interfaz automática a toda pantalla donde se mostrará pantalla del terminal remoto.

- Menú superior vertical en la parte superior.

Para servidor:

- Interfaz gráfica tanto para la autentificación por reconocimiento facial como para el form principal (excepto si la opción de invisibilidad está habilitada).

2.3 Modelo Entidad-Relación



2.4 Modelo Relación. Paso a tablas

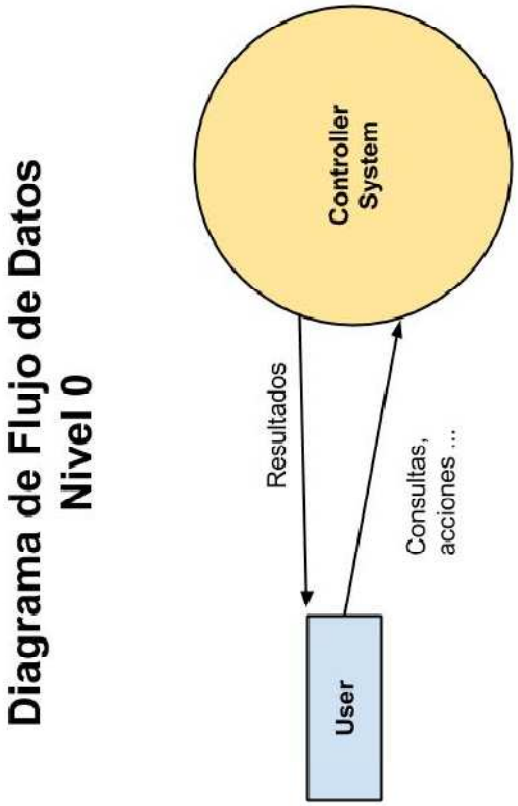
Usuario	**Webcam**	**Micrófono**	**Keylogger**
Id_User	Id_webcam	Id_micro	Id_key
Nombre	tmonitor	tmonitor	tmonitor
Apellidos	tinterval	tiempo	tiempo
User	numero	status	status
Password	status	id_user (FK)	id_user (FK)
Email	id_user (FK)		
Codserial			

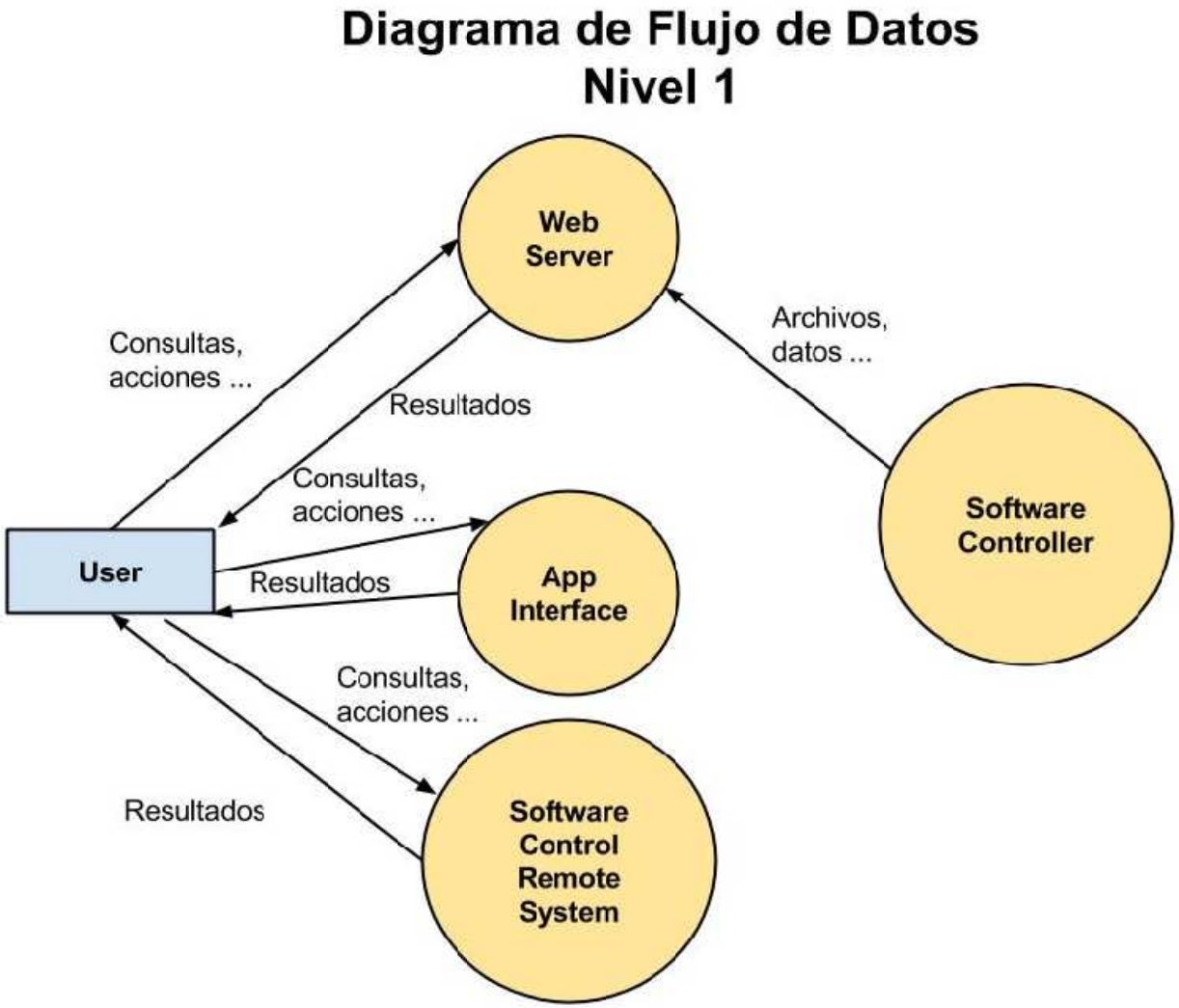
Foto	**Grabación**	**Fichero**
id_foto	id_grab	id_fich
nombre	nombre	nombre
path	path	path
fecha	fecha	fecha
id_user (FK)	id_user (FK)	id_user (FK)

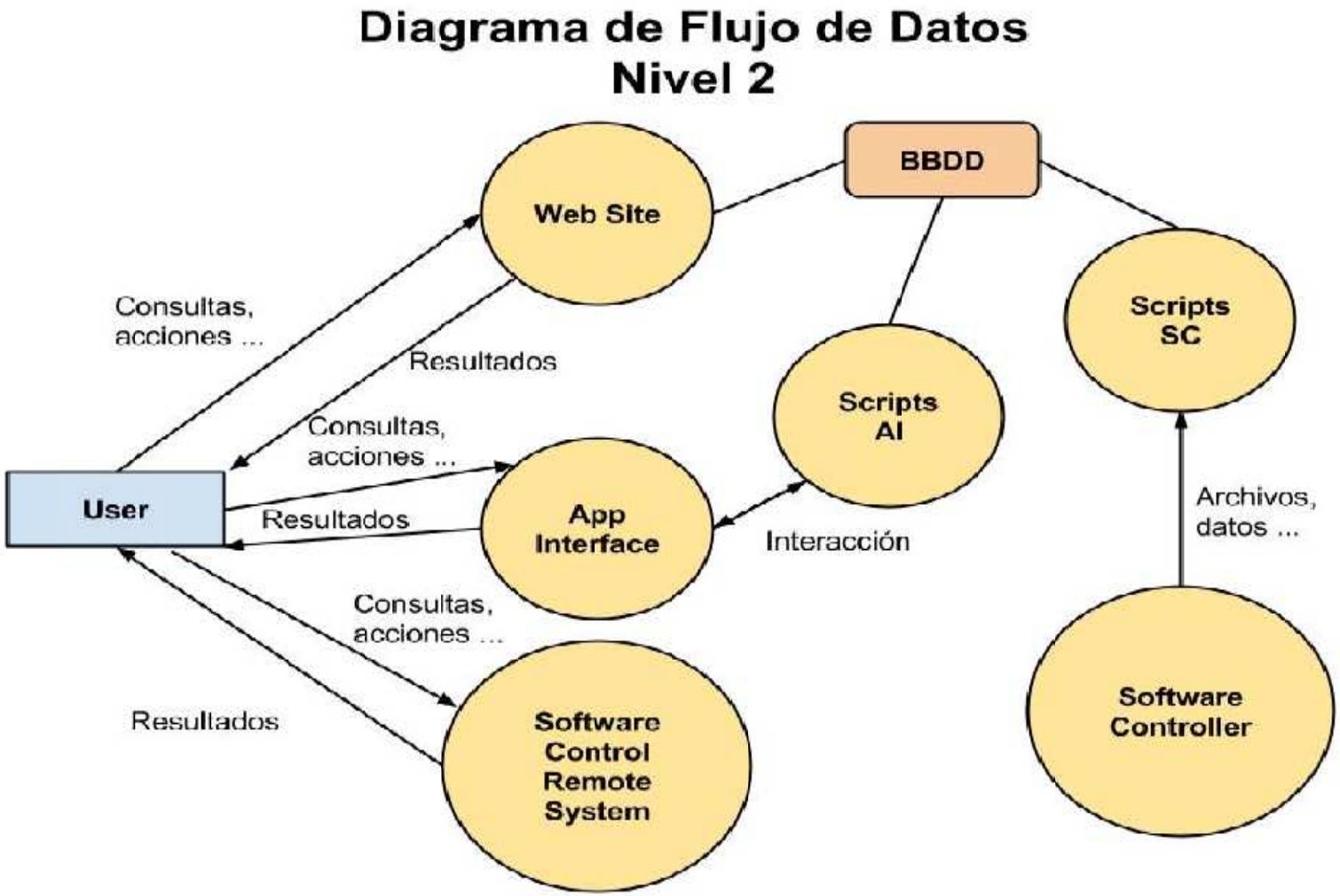
Actividad	**Compartir**
id_actividad	id_share
descripción	nombre
fecha	path
hora	fecha
tipo	tipo
id_user (FK)	destinatario
	id_user (FK)

2.5 Diagrama de flujo de datos

Con este diagrama se pretende visualizar el comportamiento del sistema con todos los elementos externos que afectan a su comportamiento. Se especificará la interacción del usuario con el software controlador, website y el usuario.

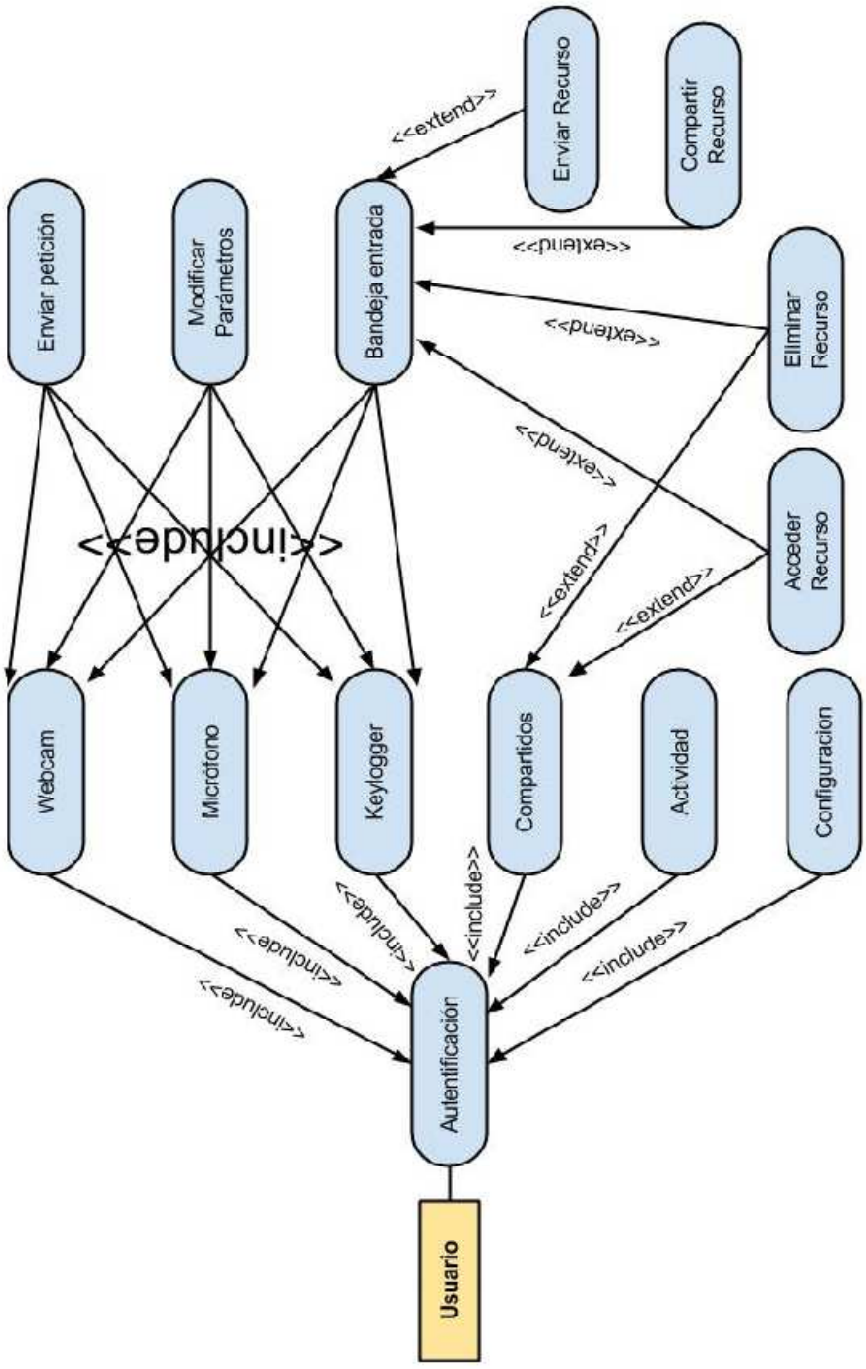






2.6 Diagrama de casos de uso

A continuación ilustraré el diagrama de casos de uso para la herramienta principal de gestión por parte del usuario en el sistema, el website.



2.7 Diagramas de clases

Diagrama de clases del software controlador

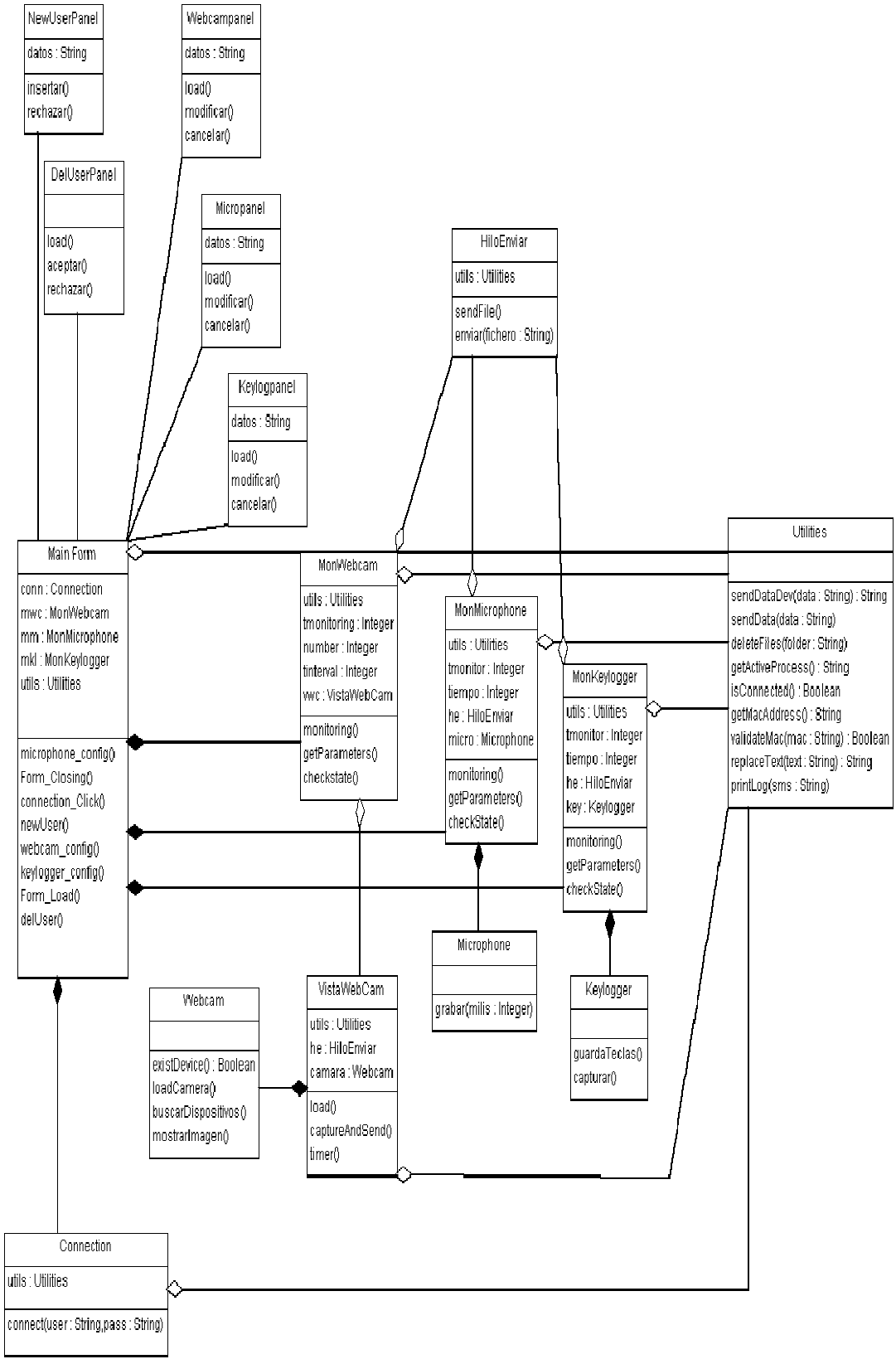


Diagrama de clases para el servidor del software Remote Control

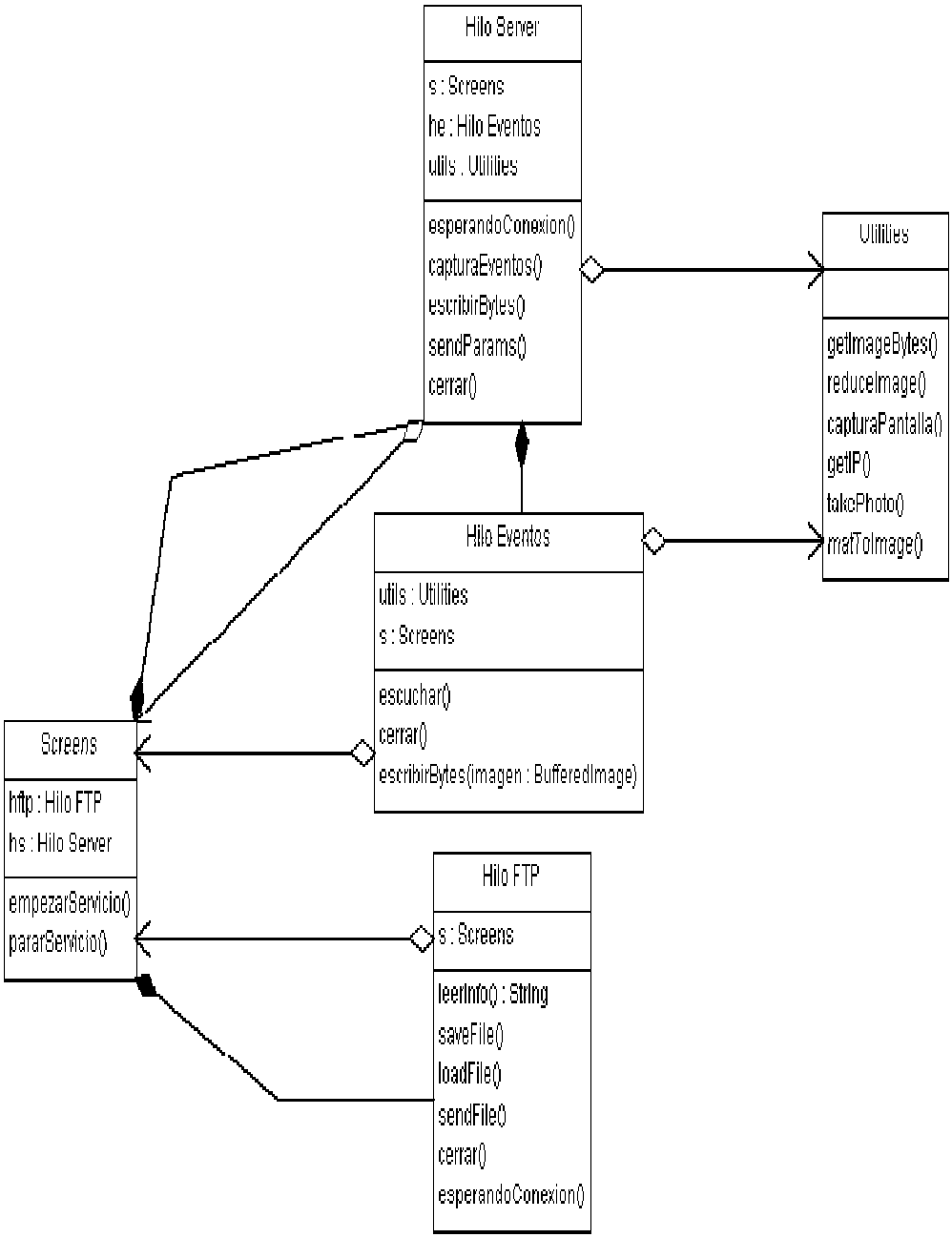


Diagrama de clases para el cliente del software Remote Control

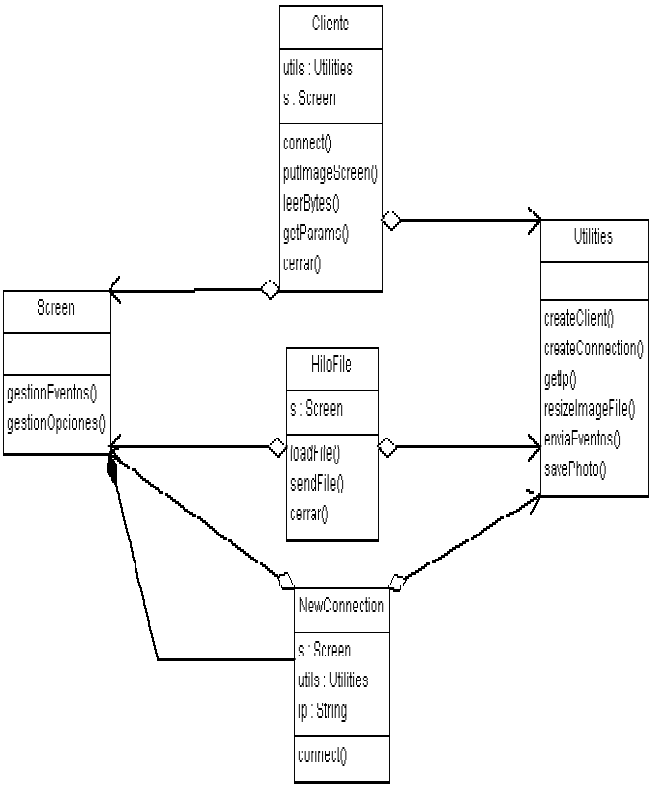
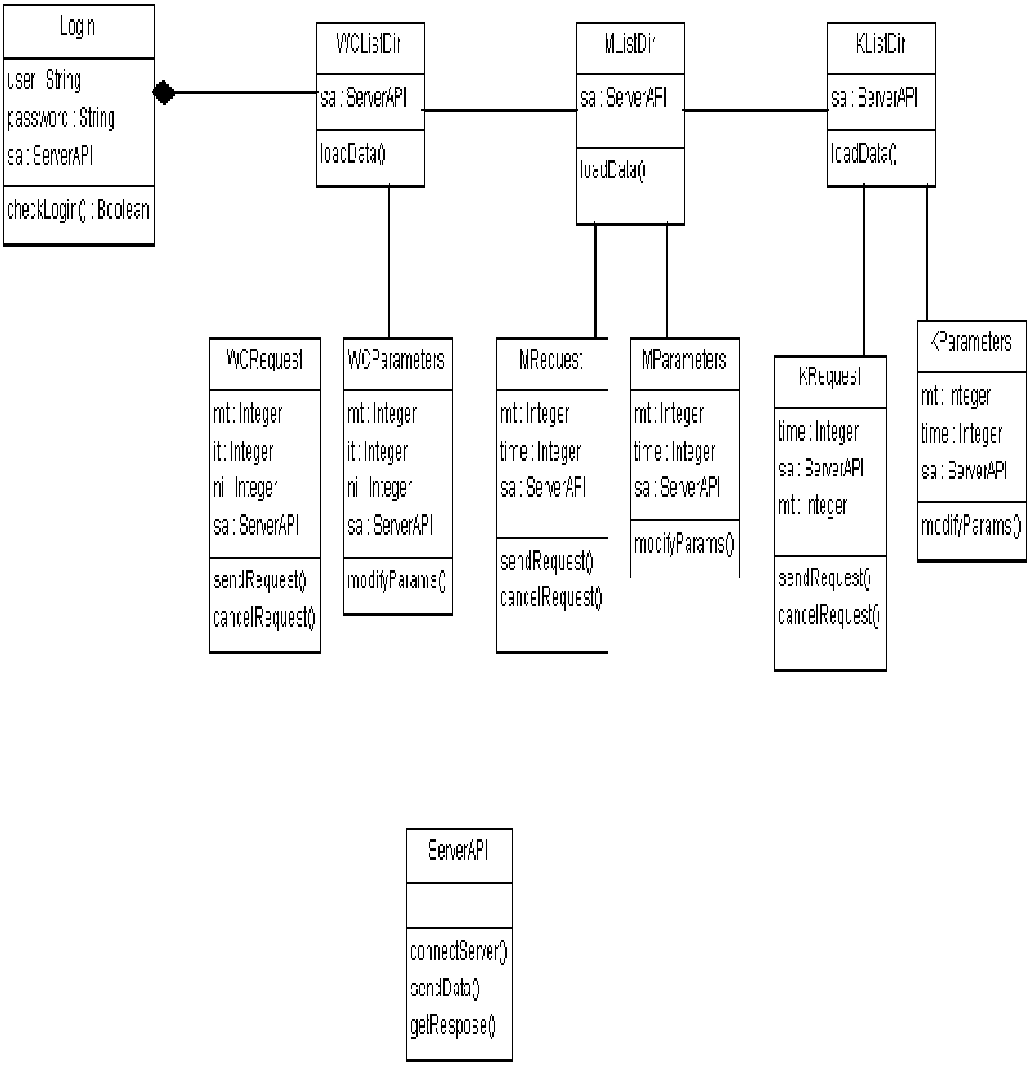


Diagrama de clases para la Android App (cliente Smartphones)



Nota: La clase ServerAPI presenta una agregación con el resto de clases. No he representado dicha relación porque no quedaría lo bastante claro gráficamente.

2.8 Especificación del parser

A continuación representaré la gramática con la que construí el analizador sintáctico para el Software Controlador con el que reconoce patrones de texto sobre terrorismo en español.

PALABRA = {Letter}+

"START SYMBOL" = <FRASE>
<FRASE> ::= <SUJETO> <VERBO> <PREDICADO> '.' | <SUJETO> <VERBO> <PREDICADO> '.' <FRASE>
<SUJETO> ::= PALABRA <SUJETO> | <NOMBRE_CLAVE> <SUJETO>
<VERBO> ::= <VBO_ACTIVAR> | <VBO_MATAR> | <VBO_DESACTIVAR> | <VBO_ASESINAR> | <VBO_PONER> | <VBO_FABRICAR> | <PPC> | <PPF> | <PA> | <FP> | <CP>
<PREDICADO> ::= PALABRA <PREDICADO> | <NOMBRE_CLAVE> <PREDICADO> | PALABRA | <NOMBRE_CLAVE>

!listado de palabras clave
<NOMBRE_CLAVE> ::= 'TERRORISMO' | 'AL QAEDA' | 'TERROR' | 'ATAQUE' | 'IRAK' | 'IRAN' | 'PAKISTAN' | 'ETA' | 'NUCLEAR' | 'DISPOSITIVO' | 'BOMBA' | 'NACIONALISMO' | 'PROTOCOLO' | 'RECLUTAMIENTO' | 'ASESINATO' | 'PAQUETE' | 'OBJETIVO' | 'CHALECO' | 'CELULA' | 'TERRORISTA' | 'ANTITERRORISMO' | 'LIBERTAD' | 'ANTITERRORISTA'
!verbo escogidos como clave
!<VERBO_INF> ::= 'MATAR' | 'ASESINAR' | 'ACTIVAR' | 'DESACTIVAR' | 'FABRICAR' | 'PONER'
!particio de los verbos
<PARTICIPIO> ::= 'MATADO' | 'ASESINADO' | 'DESACTIVADO' | 'FABRICADO' | 'PUESTO' | 'ACTIVADO'
!tiempos verbales simples (indicativo)
!presente (para todos los verbos y el generico)
<PRES_ACT> ::= 'ACTIVO' | 'ACTIVAS' | 'ACTIVA' | 'ACTIVAMOS' | 'ACTIVAIS' | 'ACTIVAN'
<PRES_MAT> ::= 'MATO' | 'MATAS' | 'MATA' | 'MATAMOS' | 'MATAIS' | 'MATAN'
<PRES_DES> ::= 'DESACTIVO' | 'DESACTIVAS' | 'DESACTIVA' | 'DESACTIVAMOS' | 'DESACTIVAIS' | 'DESACTIVAN'
<PRES_ASE> ::= 'ASESINO' | 'ASESINAS' | 'ASESINA' | 'ASESINAMOS' | 'ASESINAIS' | 'ASESINAN'
<PRES_PON> ::= 'PONGO' | 'PONES' | 'PONE' | 'PONEMOS' | 'PONEIS' | 'PONEN'
<PRES_FAB> ::= 'FABRICO' | 'FABRICAS' | 'FABRICA' | 'FABRICAMOS' | 'FABRICAIS' | 'FABRICAN'

!preterito perfecto simple (para todos los verbos y el generico)
<PPS_ACT> ::= 'ACTIVE' | 'ACTIVASTE' | 'ACTIVASTEIS' | 'ACTIVARON'
<PPS_MAT> ::= 'MATE' | 'MATASTE' | 'MATASTEIS' | 'MATARON'
<PPS_DES> ::= 'DESACTIVE' | 'DESACTIVASTE' | 'DESACTIVASTEIS' | 'DESACTIVARON'
<PPS_ASE> ::= 'ASESINE' | 'ASESINASTE' | 'ASESINASTEIS' | 'ASESINARON'
<PPS_PON> ::= 'PUSE' | 'PUSISTE' | 'PUSO' | 'PUSIMOS' | 'PUSISTEIS' | 'PUSIERON'
<PPS_FAB> ::= 'FABRIQUE' | 'FABRICASTE' | 'FABRICASTEIS' | 'FABRICARON'

!preterito imperfecto (para todos los verbos y el generico)
<PI_ACT> ::= 'ACTIVABA' | 'ACTIVABAS' | 'ACTIVABAMOS' | 'ACTIVABAIS' | 'ACTIVABAN'
<PI_MAT> ::= 'MATABA' | 'MATABAS' | 'MATABAMOS' | 'MATABAIS' | 'MATABAN'

<PI_DES> ::= 'DESACTIVABA' | 'DESACTIVABAS' | 'DESACTIVABAMOS' | 'DESACTIVABAIS' | 'DESACTIVABAN'
<PI_ASE> ::= 'ASESINABA' | 'ASESINABAS' | 'ASESINABAMOS' | 'ASESINABAIS' | 'ASESINABAN'
<PI_PON> ::= 'PONIA' | 'PONIAS' | 'PONIAMOS' | 'PONIAIS' | 'PONIAN'
<PI_FAB> ::= 'FABRICABA' | 'FABRICABAS' | 'FABRICABAMOS' | 'FABRICABAIS' | 'FABRICABAN'

!futuro (para todos los verbos y el generico)
<FUT_ACT> ::= 'ACTIVARE' | 'ACTIVARAS' | 'ACTIVARA' | 'ACTIVAREMOS' | 'ACTIVAREIS' | 'ACTIVARAN'
<FUT_MAT> ::= 'MATARE' | 'MATARAS' | 'MATARA' | 'MATAREMOS' | 'MATAREIS' | 'MATARAN'
<FUT_DES> ::= 'DESACTIVARE' | 'DESACTIVARAS' | 'DESACTIVARA' | 'DESACTIVAREMOS' | 'DESACTIVAREIS' | 'DESACTIVARAN'
<FUT_ASE> ::= 'ASESINARE' | 'ASESINARAS' | 'ASESINARA' | 'ASESINAREMOS' | 'ASESINAREIS' | 'ASESINARAN'
<FUT_PON> ::= 'PONDRE' | 'PONDRAS' | 'PONDRA' | 'PONDREMOS' | 'PONDREIS' | 'PONDRAN'
<FUT_FAB> ::= 'FABRICARE' | 'FABRICARAS' | 'FABRICARA' | 'FABRICAREMOS' | 'FABRICAREIS' | 'FABRICARAN'

!condicional (para todos los verbos y el generico)
<COND_ACT> ::= 'ACTIVARIA' | 'ACTIVARIAS' | 'ACTIVARIAMOS' | 'ACTIVARIAIS' | 'ACTIVARIAN'
<COND_MAT> ::= 'MATARIA' | 'MATARIAS' | 'MATARIAMOS' | 'MATARIAIS' | 'MATARIAN'
<COND_DES> ::= 'DESACTIVARIA' | 'DESACTIVARIAS' | 'DESACTIVARIAMOS' | 'DESACTIVARIAIS' | 'DESACTIVARIAN'
<COND_ASE> ::= 'ASESINARIA' | 'ASESINARIAS' | 'ASESINARIAMOS' | 'ASESINARIAIS' | 'ASESINARIAN'
<COND_PON> ::= 'PONDRIA' | 'PONDRIAS' | 'PONDRIAMOS' | 'PONDRIAIS' | 'PONDRIAN'
<COND_FAB> ::= 'FABRICARIA' | 'FABRICARIAS' | 'FABRICARIAMOS' | 'FABRICARIAIS' | 'FABRICARIAN'

!regla tiempos simpres
<VBO_ACTIVAR> ::= <PRES_ACT> | <PPS_ACT> | <PI_ACT> | <FUT_ACT> | <COND_ACT>
<VBO_MATAR> ::= <PRES_MAT> | <PPS_MAT> | <PI_MAT> | <FUT_MAT> | <COND_MAT>
<VBO_DESACTIVAR> ::= <PRES_DES> | <PPS_DES> | <PI_DES> | <FUT_DES> | <COND_DES>
<VBO_ASESINAR> ::= <PRES_ASE> | <PPS_ASE> | <PI_ASE> | <FUT_ASE> | <COND_ASE>
<VBO_PONER> ::= <PRES_PON> | <PPS_PON> | <PI_PON> | <FUT_PON> | <COND_PON>
<VBO_FABRICAR> ::= <PRES_FAB> | <PPS_FAB> | <PI_FAB> | <FUT_FAB> | <COND_FAB>
!Para conjugar tiempos verbales en modo indicativo
<CONJ_HABER_PRES> ::= 'HE' | 'HAS' | 'HA' | 'HEMOS' | 'HABEIS' | 'HAN'
<CONJ_HABER_PAST> ::= 'HABIA' | 'HABIAS' | 'HABIAMOS' | 'HABIAIS' | 'HABIAN'
<CONJ_HABER_ANT> ::= 'HUBE' | 'HUBISTE' | 'HUBO' | 'HUBIMOS' | 'HUBISTEIS' | 'HUBIERON'
<CONJ_HABER_FUT> ::= 'HABRE' | 'HABRAS' | 'HABRA' | 'HABREMOS' | 'HABREIS' | 'HABRAN'
<CONJ_HABER_COND> ::= 'HABRIA' | 'HABRIAS' | 'HABRIAMOS' | 'HABRIAIS' | 'HABRIAN'
!tiempos verbales compuestos indicativo
!preterito perfecto compuesto
<PPC> ::= <CONJ_HABER_PRES> <PARTICIPIO>
!preterito pluscuamperfecto
<PPF> ::= <CONJ_HABER_PAST> <PARTICIPIO>
!preterito anterior

<PA> ::= <CONJ_HABER_ANT> <PARTICIPIO>
!futuro perfecto
<FP> ::= <CONJ_HABER_FUT> <PARTICIPIO>
!condicional compuesto
<CP> ::= <CONJ_HABER_COND> <PARTICIPIO>

Hasta aquí la gramática. A continuación mostraré un ejemplo del funcionamiento del parser para una determinada cadena de entrada

AL QAEDA HA DESACTIVADO LA CELULA. EL TERRORISTA NO HABRIA MATADO AL OBJETIVO. EL CNI ACTIVARA EL PROTOCOLO DE VIGILANCIA ATAQUE.

```
=======================================
Parse Actions
=======================================

Parse Action     State   Line    Col   Parse Value                 Description
--------------  -----  -----  -----  ----------------------------  ----------------------------------------------------
Token Read         0      0      0   AL QAEDA                    Token type: AL QAEDA
Shift              1                                     The parser shifted to state 1
Token Read         1      0      9   HA                          Token type: HA
Reduce            26             (1) ::= AL QAEDA                <NOMBRE_CLAVE> ::= 'AL QAEDA'
Reduce            29             (2) ::=                       <SUJETO> ::=
Reduce            27             (3) ::= (1) (2)               <SUJETO> ::= <NOMBRE_CLAVE> <SUJETO>
Shift            108                                     The parser shifted to state 108
Token Read       108      0     12   DESACTIVADO                 Token type: DESACTIVADO
Reduce           170             (4) ::= HA                    <CONJ_HABER_PRES> ::= HA
Shift            199                                     The parser shifted to state 199
Token Read       199      0     24   LA                          Token type: PALABRA
Reduce           207             (5) ::= DESACTIVADO           <PARTICIPIO> ::= DESACTIVADO
Reduce           182             (6) ::= (4) (5)               <PPC> ::= <CONJ_HABER_PRES> <PARTICIPIO>
Reduce           196             (7) ::= (6)                   <VERBO> ::= <PPC>
Shift            208                                     The parser shifted to state 208
Token Read       208      0     27   CELULA                      Token type: CELULA
Shift              7                                     The parser shifted to state 7
Token Read         7      0     33   .                           Token type: .
Reduce           209             (8) ::= CELULA                <NOMBRE_CLAVE> ::= CELULA
Reduce           211             (9) ::= (8)                   <PREDICADO> ::= <NOMBRE_CLAVE>
Reduce           210             (10) ::= LA (9)               <PREDICADO> ::= PALABRA <PREDICADO>
Shift            213                                     The parser shifted to state 213
Token Read       213      0     35   EL                          Token type: PALABRA
Shift             18                                     The parser shifted to state 18
Token Read        18      0     38   TERRORISTA                  Token type: TERRORISTA
Shift             24                                     The parser shifted to state 24
Token Read        24      0     49   NO                          Token type: PALABRA
Reduce            26             (11) ::= TERRORISTA           <NOMBRE_CLAVE> ::= TERRORISTA
Shift             18                                     The parser shifted to state 18
Token Read        18      0     52   HABRIA                      Token type: HABRIA
Reduce            28             (12) ::=                      <SUJETO> ::=
Reduce            29             (13) ::= NO (12)              <SUJETO> ::= PALABRA <SUJETO>
Reduce            28             (14) ::= (11) (13)            <SUJETO> ::= <NOMBRE_CLAVE> <SUJETO>
Reduce            27             (15) ::= EL (14)              <SUJETO> ::= PALABRA <SUJETO>
Shift            121                                     The parser shifted to state 121
Token Read       121      0     59   MATADO                      Token type: MATADO
Reduce           167             (16) ::= HABRIA               <CONJ_HABER_COND> ::= HABRIA
Shift            201                                     The parser shifted to state 201
```

```
Token Read      201     0    66   AL                    Token type: PALABRA
Reduce          204          (17) ::= MATADO             <PARTICIPIO> ::= MATADO
Reduce          171          (18) ::= (16) (17)          <CP> ::= <CONJ_HABER_COND> <PARTICIPIO>
Reduce          196          (19) ::= (18)               <VERBO> ::= <CP>
Shift           208                                      The parser shifted to state 208
Token Read      208     0    69   OBJETIVO              Token type: OBJETIVO
Shift           16                                       The parser shifted to state 16
Token Read      16      0    77   .                     Token type: .
Reduce          209          (20) ::= OBJETIVO           <NOMBRE_CLAVE> ::= OBJETIVO
Reduce          211          (21) ::= (20)               <PREDICADO> ::= <NOMBRE_CLAVE>
Reduce          210          (22) ::= AL (21)            <PREDICADO> ::= PALABRA <PREDICADO>
Shift           213                                      The parser shifted to state 213
Token Read      213     1    0    EL                     Token type: PALABRA
Shift           18                                       The parser shifted to state 18
Token Read      18      1    3    CNI                    Token type: PALABRA
Shift           18                                       The parser shifted to state 18
Token Read      18      1    7    ACTIVARA               Token type: ACTIVARA
Reduce          28           (23) ::=                    <SUJETO> ::=
Reduce          28           (24) ::= CNI (23)           <SUJETO> ::= PALABRA <SUJETO>
Reduce          27           (25) ::= EL (24)            <SUJETO> ::= PALABRA <SUJETO>
Shift           39                                       The parser shifted to state 39
Token Read      39      1    16   EL                     Token type: PALABRA
Reduce          173          (26) ::= ACTIVARA           <FUT_ACT> ::= ACTIVARA
Reduce          192          (27) ::= (26)               <VBO_ACTIVAR> ::= <FUT_ACT>
Reduce          196          (28) ::= (27)               <VERBO> ::= <VBO_ACTIVAR>
Shift           208                                      The parser shifted to state 208
Token Read      208     1    19   PROTOCOLO             Token type: PROTOCOLO
Shift           20                                       The parser shifted to state 20
Token Read      20      1    29   DE                    Token type: PALABRA
Reduce          209          (29) ::= PROTOCOLO          <NOMBRE_CLAVE> ::= PROTOCOLO
Shift           208                                      The parser shifted to state 208
Token Read      208     1    32   VIGILANCIA            Token type: PALABRA
Shift           208                                      The parser shifted to state 208
Token Read      208     1    43   ATAQUE                Token type: ATAQUE
Shift           5                                        The parser shifted to state 5
Token Read      5       1    49   .                     Token type: .
Reduce          209          (30) ::= ATAQUE             <NOMBRE_CLAVE> ::= ATAQUE
Reduce          211          (31) ::= (30)               <PREDICADO> ::= <NOMBRE_CLAVE>
Reduce          211          (32) ::= VIGILANCIA (31)    <PREDICADO> ::= PALABRA <PREDICADO>
Reduce          212          (33) ::= DE (32)            <PREDICADO> ::= PALABRA <PREDICADO>
Reduce          211          (34) ::= (29) (33)          <PREDICADO> ::= <NOMBRE_CLAVE> <PREDICADO>
Reduce          210          (35) ::= EL (34)            <PREDICADO> ::= PALABRA <PREDICADO>
Shift           213                                      The parser shifted to state 213
Token Read      213     2    0                           Token type: EOF
Reduce          214          (36) ::= (25) (28) (35) .   <FRASE> ::= <SUJETO> <VERBO> <PREDICADO> '.'
Reduce          214               (37) ::= (15) (19) (22) . (36)   <FRASE> ::= <SUJETO> <VERBO> <PREDICADO> '.'
<FRASE>
Reduce          25           (38) ::= (3) (7) (10) . (37)   <FRASE> ::= <SUJETO> <VERBO> <PREDICADO> '.' <FRASE>
Accept          25
```

Se puede observar como el parser (LALR) va leyendo token a token y reduce cuando en la pila coincide con una regla de la gramática. Se puede ver como se trata de un parser bottom-up.

2.9 Pseudocódigo website

A continuación pondré el pseudocodigo de aquellos modulos que considere más relevantes para del website del sistema. Lo hago, debido a que la web (PHP Scripts) es desarrollado bajo un paradigma estructurado. Por lo tanto no tiene sentido realizar diagramas de clases (no es orientado a objetos).

El primer pseudocódigo se mostrará la bandeja de entrada de los recursos de un dispositivo. Con alguna que otra diferencia, pero basándose en los mismos conceptos sería para los otros dos dispositivos.

```
//identifico sesion
$id=session_de_usuario();
$html=abro_recurso();
$datos=consulta_datos_bbdd();
//resultado a mostrar
$res="";
while(getDatos($datos)){//para todos los resultados de la consulta
        $sust=replace($html,datos);
        $res=concat($res,$sust);
}
//devuelvo el resultado obtenido
devolver($res);
```

El siguiente pseudocodigo atiende a la solicitud del usuario de mandado de una orden a un dispositivo.

```
//recojo tipo de peticion (solicitar o rechazar)
$tipo=get();
if (tipo=="solicitar"){
        //compruebo que ya no hay una peticion solicitada
        $b=comprobar_solicitud();
        if(verdadero($b)){
                //modifico estado en bbdd para ofrecer solicitud
                mandar_solicitud();
        }
}
else if (tipo=="rechazar"){
        //compruebo que ya no hay una peticion rechazo
        $b=comprobar_rechazo();
        if(verdadero($b)){
                //modifico estado en bbdd para rechazar solicitud
                cancelar_solicitud();
        }

}
```

El siguiente muestra una solicitud de cambio de parametros de configuración de un dispositivo

```
//recogo parametros del usuario
datos=getData();
//compruebo si los datos son coherentes (numeros, expresiones regulares...)
$b=coherencia_data($datos);
if (verdadero($b)){
        //si son coherentes actualizo los datos
        actualiza_parametros($datos);
}
```

Los siguientes pseudocódigos que voy a mostrar son aplicados a cada uno de los recursos alojados en el servidor (imágenes, grabaciones y ficheros).

Pseudocódigo para compartir una imagen (análogamente para los otros recursos) con otros usuarios del sistema.

```
//recojo id de la imagen
id=getData();
//selecciono datos de la imagen a partir de ese id
datos=get_datos_bbdd(id);
//los guardo en variables de sesión
session_vars=save(datos);
if (existe_usuario_elegido()){//si el usuario ha elegido el usuario destino
        compartir_recurso();//modifica una tabla de la base de datos
}
else{
        //muestro pantalla selección usuario
        echo mostrar_pantalla_sel();
}
```

El siguiente pseudocódigo muestra como enviar un recurso, en este caso una imagen mediante correo electrónico

```
//recojo id de la imagen y texto a mostrar en el email
id=getID();
texto=getData();
//selecciono el path de la imagen desde la base de datos a partir del id
path=get_path_bbdd(id);
if (existe_email_destino()){
        enviar_email();
}
else{
        //muestro pantalla selección email
        echo mostrar_pantalla_email();
}
```

El siguiente pseudocódigo muestra una pantalla con un listado de la actividad generada por el usuario en el sistema.

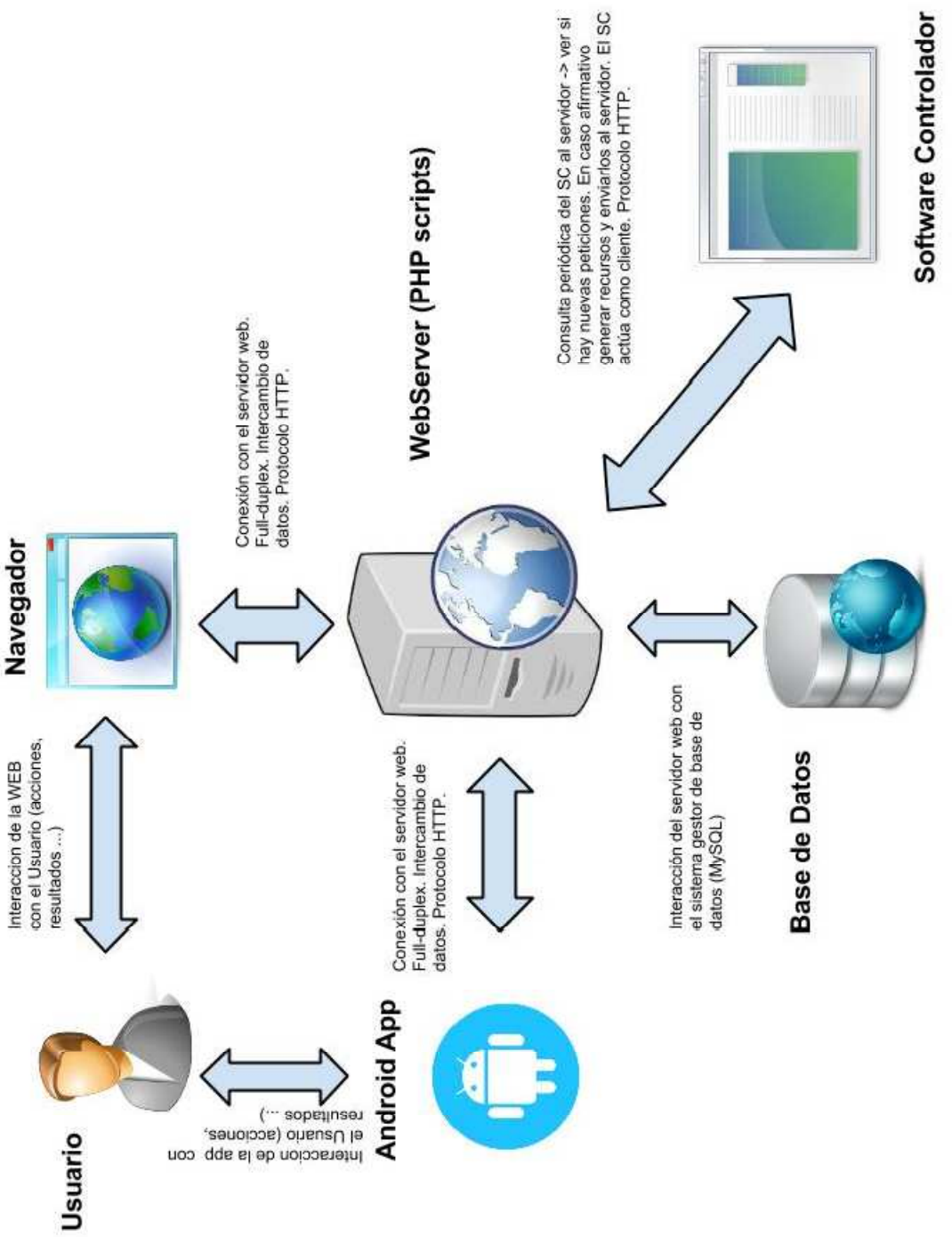
```
//extraido id usuario mediante variable de session
id=get_session_id();
//selecciono toda la actividad de el usuario ordenada por fecha y hora de forma decreciente.
actividad=get_actividad_user(id);
//muestro una pantalla con toda la actividad del usuario
echo  mostrar_actividad_usuario(actividad);
```

Por último voy a mostrar el algoritmo del panel de configuración cuando el usuario ha metido nuevos datos que desea modificar.

```
//recojo todos los datos (nombre, apellidos, usuario password, email...)
datos=getData();
//checkeo que los datos son correctos e inconsistentes:
// - Anterior contraseña válida
// - Nuevas contraseñas coinciden
// - Email forma correcta
// -...
if (checkData(datos)){
        modifico_parametros_bbdd();
}
```

2.10 Arquitectura del sistema

Con este apartado se pretende visualizar el comportamiento de todos los elementos que integran el sistema, ver cómo interactúan entre ellos y con el usuario final. En esta parte de la arquitectura queda excluida el Software Control Remoto.



Nota: La arquitectura entre Cliente y Server del software Remote Control es una conexión abierta (socket) entre ambos en el que se intercambian datos para llevar a cabo sus especificaciones. No he creído conveniente su representación debido a su simplicidad.

2.11 Metodología de trabajo

A continuación se va a mostrar la metodología de trabajo que se ha empleado para la realización del proyecto.

Lo primero de todo se planteó una propuesta justificada de proyecto para dar solución al problema planteado. Inmediatamente después se eligieron las tecnologías a usar que se consideraron para alcanzar los objetivos del proyecto.

Después, se paso a la parte de análisis y diseño, que es la parte clave para que todo el sistema funcione correctamente.

En la parte del análisis se produce la especificación mas completa del sistema que se va a implementar (semántica). Justo después la especificación de los requisitos (funcionales y no funcionales).

Luego en la parte del diseño, se hacen todos los diagramas pertinentes, diagrama de flujo de datos, de clases, de interacción, el E/R, especificaciones de gramáticas …, para que el proyectista vaya viendo y modelando perfectamente todo lo necesario para la correcta codificación del siguiente paso.

En esta parte del diseño también se incluye la estimación temporal, en la cual se estima la duración que se tardara en realizar el proyecto, y la arquitectura del sistema, donde se visualiza perfectamente el funcionamiento del sistema.

En el siguiente paso se va a llevar a cabo la implementación. Justo antes se debe elegir el lenguaje o los lenguajes en los que se va a llevar a cabo la implementación del proyecto.

En este apartado, en teoría, no se debe pensar cómo resolver conceptos del proyecto, sino que se debe codificar previamente lo que has diseñado en el apartado anterior.

A continuación se van a llevar a cabo las pruebas para comprobar que el sistema funciona correctamente.

Para ello se deben realizar las pruebas de caja negra y caja blanca (unitarias, de carga...). Las pruebas de caja negra se comprueban metiendo unos datos en el sistema a nivel de usuario y comprobando que los resultados finales que devuelve el sistema son los correctos para cualquier dato de entrada introducido. Se deben probar con todos los casos, sobre todo con los mas críticos. En cambio las pruebas de caja blanca evalúan cada parte relevante del sistema por separado introduciendo datos de entrada (incluyendo los mas críticos) y comprobando que los resultados son los esperados.

Por último se debe poner toda la parte teórica correctamente detallada, ordenada y explicita en la memoria del proyecto. La parte teórica está formada por todo lo comentado anteriormente, propuesta, análisis, diseño...

2.12 Estimación temporal

El siguiente paso a tratar es la estimacion temporal del proyecto.
Este es un apartado complejo, en el que hay evaluar cada subapartado de cada parte del desarrollo software del proyecto.

Las cinco partes en las que he dividido el proyecto son:

1. Planteamiento proyecto

Es la parte en el que la solución del problema a resolver. Tomo las decisiones pertinentes en lo que se refiere a que lenguaje de programación voy a usar, que tecnología es más conveniente … Para ello se debe consultar en diferentes fuentes, que es lo más conveniente.
Para realizar esta tarea estimo que tardare de 3 a 4 semanas.

2. Análisis y diseño del proyecto

Este apartado, en mi opinión es el apartado más importante del proyecto ya que es donde se van a tomar todas las decisiones del modelado del proyecto. Este apartado se subdivide en dos, análisis y diseño del proyecto.

En el apartado de análisis se detalla muy claramente la especificación del proyecto (semántica), especificación de requisitos (funcionales, no funcionales).
Este subapartado de análisis estará en torno a las 4-5 semanas.

El segundo subapartado es el de diseño. Aquí se realizaran todos los diagramas necesarios para visualizar las partes de las que consta el proyecto, tales como diagramas de flujo de datos, diagrama E/R, de casos de uso, de clases … A la vez en este apartado de diseño se realizara también la metodología de trabajo y la arquitectura del sistema.

Por último en este apartado también se diseñara toda la gramática correspondiente al parser de reconocimiento de patrones de texto sobre terrorismo que se implementa sobre el software controlador.

Todo este apartado de diseño podrá realizarse aproximadamente en tres meses.

3. Desarrollo e implementación del proyecto

En este apartado se va a llevar a cabo la codificación, en uno o varios, lenguajes de programación, los módulos, clases, librerías … diseñados en el apartado anterior.
Como se ha visto en la especificación de requisitos el volumen de código a desarrollar será bastante elevado, debido a la gran cantidad de material que se pretende implementar en el sistema. A eso hay que sumar el hecho de los numerosos lenguajes de programación que se llevarán a cabo para desarrollar todos los módulos del sistema y su coordinación.

Estimo que podría estar en torno a los 4 meses.

4. Pruebas

Una vez implementado el proyecto es la hora de hacer las pruebas necesarias (caja negra, caja blanca...) para evaluar el perfecto funcionamiento del proyecto. Este apartado, en mi opinión, el más impredecible de establecer su duración, ya que resulta imposible evaluar el número y la complejidad de fallos que pudiese haber en la implementación del sistema. Pero por decir algo calculo que en 6 semanas podría asegurar que todo funcionase correctamente. Pero como ya he dicho el tiempo se puede disparar o acortar. Aquí debo mencionar que aun existiría un trabajo extraordinario del proyecto si se diese el caso que necesita mantenimiento durante su ciclo de vida.

5. Memoria

En este apartado se debe ordenar, repasar, y aclarar ideas de los puntos 1 2,4. Además debe añadirse la valoración del proyecto, manual de usuarios, anexos …
Debe quedar todo perfectamente explicado.
Para esta tarea se podría hacer en 4 semanas aproximadamente.

La estimación temporal total, sumando cada una de las partes:

Planteamiento	3,5 semanas
Análisis	4,5 semanas
Diseño	9 semanas
Implementación	17 semanas
Pruebas	6 semanas (estimación que podría resultar muy variable)
Memoria	4 semanas

La duración total del proyecto estaría en torno a unas 44 semanas, es decir, unos 10 meses.

3.1 Codificación de la propuesta

En este apartado pondré el código fuente para los pseudocódigos mostrados en el anterior apartado. Quiero demostrar con esto como a partir del diseño he podido lograr mis objetivos usando una tecnología bastante común en WEB como PHP.
También quiero añadir que los pseudocódigos que elegí en su momento, para codificarlos ahora, son una representación de las distintas cosas que se usaron para la página web.

Código fuente para la bandeja de entrada de la webcam:

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
$id=getId($conectar,$serial);
$cont=1;
$nombre="thumb";
$fd=fopen("inboxwc.html","r");
while ($linea=fscanf($fd,"%[^\n]\n")){
	$cambiar="";
	$cambiar2="";
	if (stristr($linea[0],"##ini_folder##")){
		while(! (stristr($linea[0],"##fin_folder##"))){
			$linea = fscanf($fd, "%[^\n]\n");
			if (stristr($linea[0],"##ini_foto##")){
				while(! (stristr($linea[0],"##fin_foto##"))){
					$linea = fscanf($fd, "%[^\n]\n");
					$cambiar2=$cambiar2.$linea[0];
					$cambiar=$cambiar.$linea[0];
				}
			}
			else{
				$cambiar=$cambiar.$linea[0];
			}
		}
		$select="select count(*), fecha from foto where id_user='$id' group by fecha order by fecha desc";
		$query=mysql_query($select,$conectar);
		$fechahoy=date("Y-m-d");
		while($datos=mysql_fetch_array($query)){
			$lineanueva=$cambiar;
			if ($fechahoy==$datos[1]){
				$lineanueva=str_replace("##color##","red",$lineanueva);
			}
			else{
				$lineanueva=str_replace("##color##","black",$lineanueva);
			}
			$lineanueva=str_replace("##fecha##",$datos[1],$lineanueva);
```

```
                $lineanueva=str_replace("##fechfolder##",$datos[1],$lineanueva);
                if ($datos[0]<10){
        $lineanueva=str_replace("##number##"," ".$datos[0],$lineanueva);
                }
                else{
        $lineanueva=str_replace("##number##",$datos[0],$lineanueva);
                }
                $lineanueva=str_replace("##cont##",$cont,$lineanueva);
        $lineanueva=str_replace("##nomcont##",$nombre.$cont,$lineanueva);
                $f=$datos[1];
                $selfotos="select path from foto where id_user='$id' and fecha='$f'
order by path desc";
                $queryfotos=mysql_query($selfotos,$conectar);
                $fotos=mysql_fetch_array($queryfotos);
                $lineanueva=str_replace("##primerpath##",$fotos[0],$lineanueva);
                $chorizofotos="";
                while($fotos=mysql_fetch_array($queryfotos)){
                        $lineaux=$cambiar2;
                        $lineaux=str_replace("##path##",$fotos[0],$lineaux);
                        $lineaux=str_replace("##cont##",$cont,$lineaux);
                $lineaux=str_replace("##nomcont##",$nombre.$cont,$lineaux);
                        $chorizofotos=$chorizofotos.$lineaux;
                }
                $cambiar3=$cambiar2;
                $cambiar3=str_replace("##cont##",$cont,$cambiar3);
                $cambiar3=str_replace("##nomcont##",$nombre.$cont,$cambiar3);
                $lineanueva=str_replace($cambiar3,$chorizofotos,$lineanueva);
                echo $lineanueva;
                $cont=$cont+1;
        }
    }
    else{
        echo $linea[0];
    }
}
```

Código fuente para la solicitud/cancelación de una petición para la webcam al Software Controlador.

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
$id=getId($conectar,$serial);
$status=$_POST["status"];
$boton=$_POST["boton"];
$botoncal=$_POST["botoncal"];
$id=getId($conectar,$serial);
if ($boton!=""){//ha hecho peticion
	//compruebo que la opcion no este ya habilitada
	$sel="select * from webcam where id_user='$id' and status='y'";
	if (mysql_num_rows(mysql_query($sel,$conectar))>0){//si ya habia peticion
		$sel="select * from webcam where id_user='$id'";
		$datos=mysql_fetch_array(mysql_query($sel,$conectar));
		$fd=fopen("requestwc.html","r");
		while ($linea=fscanf($fd,"%[^\n]\n")){
			$lineanueva=$linea[0];
			$lineanueva=str_replace("##tmonitoring##",$datos[1],$lineanueva);
			$lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
			$lineanueva=str_replace("##number##",$datos[3],$lineanueva);
			if ($datos[4]=="n"){

	$lineanueva=str_replace("##status##","DISABLED",$lineanueva);
				$lineanueva=str_replace("##botoncal##","",$lineanueva);
			}
			else{

	$lineanueva=str_replace("##status##","ENABLED",$lineanueva);
				$lineanueva=str_replace("##botoncal##","<button class='btn btn-large btn-danger' type='submit' name='botoncal' value='botoncal'>Cancel Request</button>",$lineanueva);
			}

			echo $lineanueva;
		}
		showsms("Error sending. There is a pending request");
	}
	else{
		$update="update webcam set status='y' where id_user='$id'";
		mysql_query($update);
		$sel="select * from webcam where id_user='$id'";
		$datos=mysql_fetch_array(mysql_query($sel,$conectar));
		$fd=fopen("requestwc.html","r");
		while ($linea=fscanf($fd,"%[^\n]\n")){
```

```
                        $lineanueva=$linea[0];
                        $lineanueva=str_replace("##tmonitoring##",$datos[1],$lineanueva);
                        $lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
                        $lineanueva=str_replace("##number##",$datos[3],$lineanueva);
                        if ($datos[4]=="n"){

        $lineanueva=str_replace("##status##","DISABLED",$lineanueva);
                                $lineanueva=str_replace("##botoncal##","",$lineanueva);
                        }
                        else{

        $lineanueva=str_replace("##status##","ENABLED",$lineanueva);
                                $lineanueva=str_replace("##botoncal##","<button class='btn
btn-large btn-danger' type='submit' name='botoncal' value='botoncal'>Cancel
Request</button>",$lineanueva);
                        }

                        echo $lineanueva;
                }
                showsms("Request sent correctly");
        }
}
else{
        if ($botoncal!=""){
                $update="update webcam set status='n' where id_user='$id'";
                mysql_query($update);
                $sel="select * from webcam where id_user='$id'";
                $datos=mysql_fetch_array(mysql_query($sel,$conectar));
                $fd=fopen("requestwc.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $lineanueva=$linea[0];
                        $lineanueva=str_replace("##tmonitoring##",$datos[1],$lineanueva);
                        $lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
                        $lineanueva=str_replace("##number##",$datos[3],$lineanueva);
                        if ($datos[4]=="n"){

        $lineanueva=str_replace("##status##","DISABLED",$lineanueva);
                                $lineanueva=str_replace("##botoncal##","",$lineanueva);
                        }
                        else{

        $lineanueva=str_replace("##status##","ENABLED",$lineanueva);
                                $lineanueva=str_replace("##botoncal##","<button class='btn
btn-large btn-danger' type='submit' name='botoncal' value='botoncal'>Cancel
Request</button>",$lineanueva);
                        }

                        echo $lineanueva;
                }
```

```
                showsms("Request has been canceled");
        }
        else{
                //primera vez. Muestro datos de peticion
                $sel="select * from webcam where id_user='$id'";
                $datos=mysql_fetch_array(mysql_query($sel,$conectar));
                $fd=fopen("requestwc.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $lineanueva=$linea[0];
                        $lineanueva=str_replace("##tmonitoring##",$datos[1],$lineanueva);
                        $lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
                        $lineanueva=str_replace("##number##",$datos[3],$lineanueva);
                        if ($datos[4]=="n"){

        $lineanueva=str_replace("##status##","DISABLED",$lineanueva);
                                $lineanueva=str_replace("##botoncal##","",$lineanueva);
                        }
                        else{

        $lineanueva=str_replace("##status##","ENABLED",$lineanueva);
                                $lineanueva=str_replace("##botoncal##","<button class='btn
btn-large btn-danger' type='submit' name='botoncal' value='botoncal'>Cancel
Request</button>",$lineanueva);
                        }

                        echo $lineanueva;
                }
        }
}
```

Modificación de parámetros de la webcam:

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
$id=getId($conectar,$serial);
$tmonitoring=$_POST["tmonitoring"];
$tinterval=mysql_real_escape_string($_POST["tinterval"]);
$number=$_POST["number"];
$boton=$_POST["boton"];
if ($boton!=""){
        if (is_numeric($tinterval)){
                $update="update webcam set
tmonitor='$tmonitoring',tinterval='$tinterval',numero='$number' where id_user='$id'";
                mysql_query($update,$conectar);
                $sel="select * from webcam where id_user='$id'";
                $datos=mysql_fetch_array(mysql_query($sel,$conectar));
```

```
                $fd=fopen("parameterswc.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $lineanueva=$linea[0];
                        $lineanueva=str_replace("##".$datos[1]."##","selected",$lineanueva);
                        $lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
                        $lineanueva=str_replace("##".$datos[3]."##","selected",$lineanueva);
                        echo $lineanueva;
                }
                showsms("Parameters modified correctly");
        }
        else{
                $sel="select * from webcam where id_user='$id'";
                $datos=mysql_fetch_array(mysql_query($sel,$conectar));
                $fd=fopen("parameterswc.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $lineanueva=$linea[0];
                        $lineanueva=str_replace("##".$datos[1]."##","selected",$lineanueva);
                        $lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
                        $lineanueva=str_replace("##".$datos[3]."##","selected",$lineanueva);
                        echo $lineanueva;
                }
                showsms("Interval must be numeric");
        }

}
else{
        $sel="select * from webcam where id_user='$id'";
        $datos=mysql_fetch_array(mysql_query($sel,$conectar));
        $fd=fopen("parameterswc.html","r");
        while ($linea=fscanf($fd,"%[^\n]\n")){
                $lineanueva=$linea[0];
                $lineanueva=str_replace("##".$datos[1]."##","selected",$lineanueva);
                $lineanueva=str_replace("##tinterval##",$datos[2],$lineanueva);
                $lineanueva=str_replace("##".$datos[3]."##","selected",$lineanueva);
                echo $lineanueva;
        }
}
```

Código para compartir cualquier recurso con otro usuario del sistema.

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
$idfile=$_GET["id"];
$tipo=$_GET["type"];
//valores recibidos de sharedfilter.html
$userfilter=mysql_real_escape_string($_POST["userfilter"]);
$botonfilterini=$_POST["botonfilterini"];
//valores recibidos de sharedchoose.html
$userchoose=$_POST["userchoose"];
$botonchoose=$_POST["botonchoose"];
if (isset($serial)){
	$id=getId($conectar,$serial);
	if ($idfile!=""){
		if ($tipo=="1"){
			$sel="select * from foto where id_foto='$idfile'";
			$datos=mysql_fetch_array(mysql_query($sel,$conectar));
			$enlace="daemon/".$datos[2];
		}
		else if ($tipo=="2"){
			$sel="select * from grabacion where id_grab='$idfile'";
			$datos=mysql_fetch_array(mysql_query($sel,$conectar));
			$enlace="img/soundicon3.png";
		}
		else if ($tipo=="3"){
			$sel="select * from fichero where id_fich='$idfile'";
			$datos=mysql_fetch_array(mysql_query($sel,$conectar));
			$enlace="img/pdficon.png";
		}

		$fd=fopen("sharedfilter.html","r");

		while ($linea=fscanf($fd,"%[^\n]\n")){
			$lineanueva=$linea[0];
			$lineanueva=str_replace("##enlace##",$enlace,$lineanueva);
			$lineanueva=str_replace("##nombre##",$datos[1],$lineanueva);
			$lineanueva=str_replace("##fecha##",$datos[3],$lineanueva);
			echo $lineanueva;
		}
		$_SESSION["tiposhare"]=$tipo;
		$_SESSION["idshare"]=$datos[0];
		$_SESSION["enlaceshare"]=$datos[2];
		$_SESSION["iconshare"]=$enlace;
		$_SESSION["nombreshare"]=$datos[1];
		$_SESSION["fechashare"]=$datos[3];
```

```
}
if ($botonfilterini!=""){
        if ($userfilter==""){
                $fd=fopen("sharedfilter.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $lineanueva=$linea[0];

$lineanueva=str_replace("##enlace##",$_SESSION["iconshare"],$lineanueva);

$lineanueva=str_replace("##nombre##",$_SESSION["nombreshare"],$lineanueva);

$lineanueva=str_replace("##fecha##",$_SESSION["fechashare"],$lineanueva);
                        echo $lineanueva;
                }
                showsms("The field filter of users can not be empty");
        }
        else{
                $fd=fopen("sharedchoose.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $cambiar="";
                        if (stristr($linea[0],"##ini_user##")){
                                while(! (stristr($linea[0],"##fin_user##"))){
                                        $linea = fscanf($fd, "%[^\n]\n");
                                        $cambiar=$cambiar.$linea[0];
                                }
                                $iduser=getId($conectar,$serial);
                                $select="select id_user, user from usuario where (nombre like '%".$userfilter."%' or apellidos like '%".$userfilter."%' or user like '%".$userfilter."%') and id_user <> '$iduser'";
                                $query=mysql_query($select,$conectar);
                                while($datos=mysql_fetch_array($query)){
                                        $lineanueva=$cambiar;

$lineanueva=str_replace("##id_user##",$datos[0],$lineanueva);

$lineanueva=str_replace("##nombreuser##",$datos[1],$lineanueva);
                                        echo $lineanueva;
                                }
                        }
                        else{
                                $lineanueva=$linea[0];

$lineanueva=str_replace("##enlace##",$_SESSION["iconshare"],$lineanueva);

$lineanueva=str_replace("##nombre##",$_SESSION["nombreshare"],$lineanueva);

$lineanueva=str_replace("##fecha##",$_SESSION["fechashare"],$lineanueva);
                                echo $lineanueva;
                        }
```

```
                        }
                        $_SESSION["userfilter"]=$userfilter;
                }
        }
        if ($botonchoose!=""){
                $userfilter=$_SESSION["userfilter"];
                if ($userchoose!=""){
                        $idf=$_SESSION["idshare"];
                        $enlace=$_SESSION["enlaceshare"];
                        $icon=$_SESSION["iconshare"];
                        $nombre=$_SESSION["nombreshare"];
                        $fecha=$_SESSION["fechashare"];
                        $tipo=$_SESSION["tiposhare"];
                        $insert="insert into share values
(NULL,'$nombre','$enlace','$fecha','$tipo','$userchoose','$id')";
                        mysql_query($insert);
                        $fd=fopen("sharedchoose.html","r");
                        while ($linea=fscanf($fd,"%[^\n]\n")){
                                $cambiar="";
                                if (stristr($linea[0],"##ini_user##")){
                                        while(! (stristr($linea[0],"##fin_user##"))){
                                                $linea = fscanf($fd, "%[^\n]\n");
                                                $cambiar=$cambiar.$linea[0];
                                        }
                                        $select="select id_user, user from usuario where
nombre like '%".$userfilter."%' or apellidos like '%".$userfilter."%' or user like
'%".$userfilter."%'";
                                        $query=mysql_query($select,$conectar);
                                        while($datos=mysql_fetch_array($query)){
                                                $lineanueva=$cambiar;

        $lineanueva=str_replace("##id_user##",$datos[0],$lineanueva);

        $lineanueva=str_replace("##nombreuser##",$datos[1],$lineanueva);
                                                echo $lineanueva;
                                        }
                                }
                                else{
                                        $lineanueva=$linea[0];

        $lineanueva=str_replace("##enlace##",$icon,$lineanueva);

        $lineanueva=str_replace("##nombre##",$_SESSION["nombreshare"],$lineanueva);

        $lineanueva=str_replace("##fecha##",$_SESSION["fechashare"],$lineanueva);
                                        echo $lineanueva;
                                }
                        }
                        showsms('File shared correctly');
```

```
                }
                else{
                        $fd=fopen("sharedchoose.html","r");
                        while ($linea=fscanf($fd,"%[^\n]\n")){
                                $cambiar="";
                                if (stristr($linea[0],"##ini_user##")){
                                        while(! (stristr($linea[0],"##fin_user##"))){
                                                $linea = fscanf($fd, "%[^\n]\n");
                                                $cambiar=$cambiar.$linea[0];
                                        }
                                        $select="select id_user, user from usuario where nombre like '%".$userfilter."%' or apellidos like '%".$userfilter."%' or user like '%".$userfilter."%'";
                                        $query=mysql_query($select,$conectar);
                                        while($datos=mysql_fetch_array($query)){
                                                $lineanueva=$cambiar;

        $lineanueva=str_replace("##id_user##",$datos[0],$lineanueva);

        $lineanueva=str_replace("##nombreuser##",$datos[1],$lineanueva);
                                                echo $lineanueva;
                                        }
                                }
                                else{
                                        $lineanueva=$linea[0];

        $lineanueva=str_replace("##enlace##",$icon,$lineanueva);

        $lineanueva=str_replace("##nombre##",$_SESSION["nombreshare"],$lineanueva);

        $lineanueva=str_replace("##fecha##",$_SESSION["fechashare"],$lineanueva);
                                        echo $lineanueva;
                                }
                        }
                        showsms('Any user choosen');
                }
        }
}
else{
        mostrarlogin();
}
```

Código fuente para enviar por email un recurso:

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
$idfile=$_GET["id"];
$tipo=$_GET["type"];
$sms=$_POST["smsop"];
$email=mysql_real_escape_string($_POST["destemail"]);
$emailsel=$_POST["destemailsel"];
$boton=$_POST["boton"];
if (isset($serial)){
        if ($idfile!=""){
                if ($tipo=="1"){
                        $sel="select * from foto where id_foto='$idfile'";
                        $datos=mysql_fetch_array(mysql_query($sel,$conectar));
                        $enlace="daemon/".$datos[2];
                        $pathaux="daemon/".$datos[2];
                }
                else if ($tipo=="2"){
                        $sel="select * from grabacion where id_grab='$idfile'";
                        $datos=mysql_fetch_array(mysql_query($sel,$conectar));
                        $enlace="img/soundicon3.png";
                        $pathaux="daemon/".$datos[2];
                }
                else if ($tipo=="3"){
                        $sel="select * from fichero where id_fich='$idfile'";
                        $datos=mysql_fetch_array(mysql_query($sel,$conectar));
                        $enlace="img/pdficon.png";
                        $pathaux="daemon/".$datos[2];
                }

                $fd=fopen("sendfile.html","r");
                while ($linea=fscanf($fd,"%[^\n]\n")){
                        $cambiar="";
                        if (stristr($linea[0],"##ini_option##")){
                                while(! (stristr($linea[0],"##fin_option##"))){
                                        $linea = fscanf($fd, "%[^\n]\n");
                                        $cambiar=$cambiar.$linea[0];
                                }
                                $lineanueva=$cambiar;
                                $lineanueva=str_replace("##val##","",$lineanueva);
                                $lineanueva=str_replace("##correo##","Look                    for
User",$lineanueva);
                                echo $lineanueva;
                        }
                        else{
```

```
                        $lineanueva=$linea[0];
                        $lineanueva=str_replace("##enlace##",$enlace,$lineanueva);

        $lineanueva=str_replace("##nombre##",$datos[1],$lineanueva);
                        $lineanueva=str_replace("##fecha##",$datos[3],$lineanueva);
                        echo $lineanueva;
                    }
                }
                $_SESSION["iconmail"]=$enlace;
                $_SESSION["pathmail"]=$pathaux;
                $_SESSION["nombremail"]=$datos[1];
                $_SESSION["fechamail"]=$datos[3];
        }
        else{//han pulsado Send (enviado formulario post)
                if (validaEmail($email)==1){

        $b=enviarEmail(getId($conectar,$serial),$email,$_SESSION["pathmail"],$_SESSION["no
mbremail"],$sms,$conectar);
                    if ($b){
                        $sms="Email sent correctly";
                    }
                    else{
                        $sms="There is an error sending the email. Please, try again";
                    }
                    $fd=fopen("sendfile.html","r");
                    while ($linea=fscanf($fd,"%[^\n]\n")){
                        $cambiar="";
                        if (stristr($linea[0],"##ini_option##")){
                            while(! (stristr($linea[0],"##fin_option##"))){
                                $linea = fscanf($fd, "%[^\n]\n");
                                $cambiar=$cambiar.$linea[0];
                            }
                            $lineanueva=$cambiar;
                            $lineanueva=str_replace("##val##","",$lineanueva);
                            $lineanueva=str_replace("##correo##","Look for
User",$lineanueva);
                            echo $lineanueva;
                        }
                        else{
                            $lineanueva=$linea[0];

        $lineanueva=str_replace("##enlace##",$_SESSION["iconmail"],$lineanueva);

        $lineanueva=str_replace("##nombre##",$_SESSION["nombremail"],$lineanueva);

        $lineanueva=str_replace("##fecha##",$_SESSION["fechamail"],$lineanueva);
                            echo $lineanueva;
                        }
                    }
```

```
                showsms($sms);
            }
            else{
                if (validaEmail($emailsel)==1){

    $b=enviarEmail(getId($conectar,$serial),$emailsel,$_SESSION["pathmail"],$_SESSION["
nombremail"],$sms,$conectar);

                        if ($b){
                            $sms="Email sent correctly";
                        }
                        else{
                            $sms="There is an error sending the email. Please, try
again";
                        }
                        $fd=fopen("sendfile.html","r");
                        while ($linea=fscanf($fd,"%[^\n]\n")){
                            $cambiar="";
                            if (stristr($linea[0],"##ini_option##")){
                                while(! (stristr($linea[0],"##fin_option##"))){
                                    $linea = fscanf($fd, "%[^\n]\n");
                                    $cambiar=$cambiar.$linea[0];
                                }
                                $lineanueva=$cambiar;

    $lineanueva=str_replace("##val##","",$lineanueva);
                                $lineanueva=str_replace("##correo##","Look
for User",$lineanueva);
                                echo $lineanueva;
                            }
                            else{
                                $lineanueva=$linea[0];

    $lineanueva=str_replace("##enlace##",$_SESSION["iconmail"],$lineanueva);

    $lineanueva=str_replace("##nombre##",$_SESSION["nombremail"],$lineanueva);

    $lineanueva=str_replace("##fecha##",$_SESSION["fechamail"],$lineanueva);
                                echo $lineanueva;
                            }
                        }
                        showsms($sms);
                }
                else{
                    //aplico filtro de busqueda
                    $fd=fopen("sendfile.html","r");
                    while ($linea=fscanf($fd,"%[^\n]\n")){
                        $cambiar="";
                        if (stristr($linea[0],"##ini_option##")){
```

```
                                        while(! (stristr($linea[0],"##fin_option##"))){
                                                $linea = fscanf($fd, "%[^\n]\n");
                                                $cambiar=$cambiar.$linea[0];
                                        }
                                        $iduser=getId($conectar,$serial);
                                        $select="select user,email from usuario where (nombre like '%".$emailsel."%' or apellidos like '%".$emailsel."%' or user like '%".$emailsel."%' or email like '%".$emailsel."%') and email <> '' and id_user <> '$iduser'";
                                        $query=mysql_query($select,$conectar);
                                        while($datos=mysql_fetch_array($query)){
                                                $lineanueva=$cambiar;

        $lineanueva=str_replace("##val##",$datos[1],$lineanueva);

        $lineanueva=str_replace("##correo##",$datos[0],$lineanueva);
                                                echo $lineanueva;
                                        }
                                }
                                else{
                                        $lineanueva=$linea[0];

        $lineanueva=str_replace("##enlace##",$_SESSION["iconmail"],$lineanueva);

        $lineanueva=str_replace("##nombre##",$_SESSION["nombremail"],$lineanueva);

        $lineanueva=str_replace("##fecha##",$_SESSION["fechamail"],$lineanueva);
                                                echo $lineanueva;
                                        }
                                }
                        }

                }
        }

}
else{
        mostrarlogin();
}
```

Código fuente para mostrar la actividad generado por el usuario en el sistema.

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
if (isset($serial)){
	$id=getId($conectar,$serial);
	$fd=fopen("activity.html","r");
	while ($linea=fscanf($fd,"%[^\n]\n")){
		$cambiar="";
		if (stristr($linea[0],"##ini_activity##")){
			while(! (stristr($linea[0],"##fin_activity##"))){
				$linea = fscanf($fd, "%[^\n]\n");
				$cambiar=$cambiar.$linea[0];
			}
			$select="select description,fecha,hora,tipo from activity where
id_user='$id' order by id desc";
			$query=mysql_query($select,$conectar);
			$noprint=true;
			while($data=mysql_fetch_array($query)){
				$fechauxant=$fechaux;
				$fechaux=$data[1];
				if (! stristr($fechauxant,$fechaux)){
					if (! $noprint){
						echo '<tr><td colspan="2"><hr /></td></tr>';
					}
					else{
						$noprint=false;
					}
				}
				$lineanueva=$cambiar;
				if ($data[3]=="1"){

	$lineanueva=str_replace("##tipo##","imagefile1.png",$lineanueva);
				}
				else if ($data[3]=="2"){

	$lineanueva=str_replace("##tipo##","soundicon3.png",$lineanueva);
				}
				else if ($data[3]=="3"){

	$lineanueva=str_replace("##tipo##","pdficon.png",$lineanueva);
				}
				else if ($data[3]=="4"){

	$lineanueva=str_replace("##tipo##","delete.png",$lineanueva);
				}
```

```
                                        else if ($data[3]=="5"){

        $lineanueva=str_replace("##tipo##","alarma.gif",$lineanueva);
                                        }
                                        else if ($data[3]=="6"){

        $lineanueva=str_replace("##tipo##","alarma.gif",$lineanueva);
                                        }

        $lineanueva=str_replace("##description##",$data[0],$lineanueva);
                                        $lineanueva=str_replace("##fecha##",$data[1],$lineanueva);
                                        $lineanueva=str_replace("##hora##",$data[2],$lineanueva);

                                        echo $lineanueva;

                                }
                        }
                        else{
                                echo $linea[0];
                        }
        }
}
else{
        mostrarlogin();
}
```

Por último mostraré el código fuente que se encarga de la modificación de parámetros personales del usuario en el sistema:

```
include 'functions.php';
session_start();
$serial=$_SESSION["codserial"];
$conectar=connectBBDD();
$nombre=mysql_real_escape_string($_POST["nombre"]);
$apellidos=mysql_real_escape_string($_POST["apellidos"]);
$email=mysql_real_escape_string($_POST["email"]);
$user=mysql_real_escape_string($_POST["user"]);
$pass=mysql_real_escape_string($_POST["pass"]);
$newpass=mysql_real_escape_string($_POST["newpass"]);
$repnewpass=mysql_real_escape_string($_POST["repnewpass"]);
$boton=mysql_real_escape_string($_POST["boton"]);
if (isset($serial)){
        if ($boton!=""){
                $b=false;
                if (($pass=="") && ($newpass=="") && ($repnewpass="")){
                        if ($email!=""){
                                if (validaEmail($email)==1){
```

```
                                        $update="update usuario set nombre='$nombre',
apellidos='$apellidos', user='$user', email='$email' where codserial='$serial'";
                                        $b=true;
                                }
                                else{
                                        $sms="The email is incorrect";
                                }
                        }
                        else{
                                $update="update usuario set nombre='$nombre',
apellidos='$apellidos', user='$user', email='' where codserial='$serial'";
                                $b=true;
                        }
                }
                else{
                        $sel="select * from usuario where password=password('$pass') and
codserial='$serial'";
                        if (mysql_num_rows(mysql_query($sel,$conectar))>0){
                                if (stristr($newpass,$repnewpass)){
                                        $update="update usuario set nombre='$nombre',
apellidos='$apellidos',        user='$user',        password=password('$newpass')        where
codserial='$serial'";
                                        $b=true;
                                }
                                else{
                                        $sms="New passwords are not the same";
                                }
                        }
                        else{
                                $sms="The old password is not correct";

                        }
                }
                if ($b){
                        mysql_query($update);
                        $sms="Data changed correctly";
                }
                mostrarconfig($serial,$conectar);
                showsms($sms);

        }
        else{
                mostrarconfig($serial,$conectar);
        }
}
else{
        mostrarlogin();
}
```

3.2 Problemas que han surgido

En este punto voy a ilustrar los problemas que han surgido durante el análisis, diseño y desarrollo del proyecto.

Voy a dividir los problemas conforme a las diferentes partes que definen el sistema.

En primer lugar me voy a centrar en el Software Controller. La primera cuestión que me surgió fue cuando decidí que el software debía tener una interfaz gráfica para una utilización más clara y sencilla por parte del usuario. Siempre es más cómodo para un usuario standar utilizar cuadros de texto, cajas de diálogos, menús, botones … que no comandos sobre un terminal.

Otro punto que tuve que tener en cuenta fue el hecho de minimizar los retardos a la hora de subir recursos (imagenes, grabaciones, ficheros, strings …) al servidor web. Esto es debido a que ciertos servidores (gratuitos) no te dejan cambiar los parámetros de configuración, en este caso el timeout de subida de ficheros.

Sobre los módulos del Software Controlador para el parser sobre terrorismo se me plantearon varios problemas y cuestiones. Sobre todo el hecho de como llevar a cabo el algoritmo y como codificarlo para integrarlo en el software (recuerdo que está hecho con .NET con el lenguaje C#).

Otro punto, bastante importante, fue el hecho de como coordinar las acciones llevadas a cabo por el software controlador con el servidor web (junto con la base de datos). Todo transacción o cambio debe ser reflejado en el sistema para que el usuario pueda verlos.

A la hora de generar los recursos que ofrece el sistema (imágenes, grabaciones y captura de teclas), deben ser en formatos lo más estandar posibles. Para que el usuario final con las aplicaciones y programas más estandar puedan ser fácilmente accesibles. Obviamente se deben importar o desarrollar librerías para cada una de estas funciones.

Resulta obvio que el software controlador debe ser totalmente robusto en situaciones que el usuario pueda ocasionar (errores del usuario) o en situaciones que se den por fallos en la infraestructura (problemas de red, retardos...) Ante todo eso el programa debe seguir funcionando iterativamente y notificar en la medida de lo posible los errores que han podido suceder.

Por último uno de los problemas de seguridad más grandes que tuve fue con el sistema de autentificación del software controlador. Dicho acceso funciona mediante un usuario y contraseña válidos. Pero resulta que para un ordenador concreto lo lógico es pensar que solo se pueda acceder con un usuario y contraseña validos (cualquier par de usuario y contraseña válidos no debe poder validar correctamente el software en cualquier ordenador). Por eso es por lo que tuve que generar un código del cual se obtuviera un identificador ÚNICO que validara a uno y solo a un par de usuario y contraseña válidos.

A partir de aquí detallaré los problemas más importantes que tuve con la página web.

Uno de los principales fue la de tener que desarrollar un interfaz responsive que se ajustara y visualizara perfectamente con todos los tipos de terminales y navegadores.

Otro de los incovenientes fue el hecho de incorporar un sistema de galerías lo más rapida y atractiva posible para el usuario para ver las imagenes producidas por la web cam. Así mismo con las grabaciones provenientes del micrófono debían ser bien reproducidas por cualquier navegador.

El sistema ofrece varias funcionalidades, concretamente en la web, donde es necesario enviar correos electrónicos al usuario. Se debe abordar este tema de la forma más funcional y amigable.

Se debe estructurar de manera clara y sencilla todas las funcionalidades que ofrece cada uno de los dispositivos diferenciandolos en zonas claramente distintas. Debe quedar bastante diferenciado cada zona.

Del software auxiliar Remote Control tenemos que tener en cuenta varios problemas y cuestiones.

De la parte del server hay que abordar el tema de la autentificación. Recuerdo que es mediante reconocimiento facial. Se trata de utilizar un conjunto de librerías para generar herramientas que permitan la detección, el entrenamiento y el posterior reconocimiento facial. Esto permitirá la autenticación para determinar el acceso o rechazo al ServerRemote.

En el servidor debe poder abortar en el momento que desee la conexión establecida por la otra parte (A continuación describiré como en el cliente la desconexión no implica el cierre del server).

La funcionalidad de la transmisión de ficheros debe ser totalmente concurrente con la función principal del software. Ninguna de las dos funcionalidades debe inturrumpir a la otra.

En la parte del cliente se debe poder ofrecer la posibilidad de generar un informe con los datos de la conexión, siempre y cuando haya sido finalizada previamente (por parte del ciente o por parte del servidor). Una de las estadísticas que se muestran en el informe será representada en un gráfico. Por lo tanto también se debera importar alguna librería para desarrollar dicha funcionalidad.

Al igual que en el server la funcionalidad principal y la transmisión de ficheros deben ser concurrentes. Cuando el cliente cierra la conexión no necesariamente implica que se deba cerrar el el servicio del server.

En cuanto a la aplicación Android, el principal problema, al igual que lo interfaz para la web es el hecho de diferenciar claramente cada uno de los dispositivos y dentro de cada uno, todas las opciones.

Se me planteó un problema también a la hora de la autentificación. Funciona igual que en la web, con usuario y contraseña. Pero para una mayor agilidad de la aplicación se pedirán las credenciales la primera vez que se ejecute la aplicación y la primera vez después de cada desconexión (opción que puede utilizar el usuario).Por lo tanto se debe implementar un mecanismo que permita la autentificación automática.

3.3 Solución a los problemas planteados

En este apartado voy a dar paso a las soluciones adoptadas para los problemas planteados en el apartado anterior.

Voy a empezar con los problemas surgidos en el software controlador.

Para la interfaz gráfica del software controlador decidí abordarla con el framework .NET con el lenguaje de programación C#. Para ello utilizé el IDE proporcionado por microsoft llamado Microsoft Visual C# 2008. Aproveché todas las ventajas de .NET (con Windows Forms) y la simplicidad de un lenguaje orientado a objetos como C#, que es potente pero sencillo.
Tiene otras desventajas, como el hecho de que solo funciona para el sistema operativo Windows, de Microsoft.

Tecnología .NET (C#)

El paradigma orientado a objetos está abarcado por numerosas tecnologías (Java, C++, Python, C++...). Una de las más usadas está diseñada por Microsoft, el framework .NET con el lenguaje de programación C#. A continuación explicaré más detalladamente que en el punto 1.2 esta tecnología. Documentación obtenida de la página web oficial de Microsoft [5].

.NET Framework es un entorno de ejecución administrado que proporciona diversos servicios a las aplicaciones en ejecución. Consta de dos componentes principales: *Common Language Runtime* (CLR), que es el motor de ejecución que controla las aplicaciones en ejecución, y la biblioteca de clases de .NET Framework, que proporciona una biblioteca de código probado y reutilizable al que pueden llamar los desarrolladores desde sus propias aplicaciones. Los servicios que ofrece .NET Framework a las aplicaciones en ejecución son los siguientes:

- Administración de la memoria. En muchos lenguajes de programación, los programadores son responsables de asignar y liberar memoria y de administrar la vida útil de los objetos. En las aplicaciones de .NET Framework, CLR proporciona estos servicios en nombre de la aplicación.

- Sistema de tipos comunes. En los lenguajes de programación tradicionales, el compilador define los tipos básicos, lo que complica la interoperabilidad entre lenguajes.En .NET Framework, los tipos básicos los define el sistema de tipos de .NET Framework y son comunes a todos los lenguajes que tienen como destino .NET Framework.

- Biblioteca de clases extensa. En lugar de tener que escribir cantidades extensas de código para controlar operaciones comunes de programación de bajo nivel, los programadores pueden usar una biblioteca de tipos accesible en todo momento y sus miembros desde la biblioteca de clases de .NET Framework.

- Frameworks y tecnologías de desarrollo. .NET Framework incluye bibliotecas para determinadas áreas de desarrollo de aplicaciones, como ASP.NET para aplicaciones web, ADO.NET para el acceso a los datos y *Windows Communication Foundation* para las aplicaciones orientadas a servicios.

- Interoperabilidad de lenguajes. Los compiladores de lenguajes cuya plataforma de destino es .NET Framework emiten un código intermedio denominado Lenguaje intermedio común (CIL), que, a su vez, se compila en tiempo de ejecución a través de *Common Language Runtime*. Con esta característica, las rutinas escritas en un lenguaje están accesibles a otros lenguajes, y los programadores pueden centrarse en crear aplicaciones en su lenguaje o lenguajes preferidos.

- Compatibilidad de versiones. Con raras excepciones, las aplicaciones que se desarrollan con una versión determinada de .NET Framework se pueden ejecutar sin modificaciones en una versión posterior.

- Ejecución en paralelo. .NET Framework ayuda a resolver conflictos entre versiones y permite que varias versiones de *Common Language Runtime* coexistan en el mismo equipo. Esto significa que también pueden coexistir varias versiones de las aplicaciones, y que una aplicación se puede ejecutar en la versión de .NET Framework con la que se compiló.

- Compatibilidad con múltiples versiones. Al usar la Biblioteca de clases portable de .NET Framework, los desarrolladores pueden crear programas que funcionen en varias plataformas de .NET Framework, como Windows 7, Windows 8, Windows Phone...

En cuanto al lenguaje de programación para utilizar el Framework .NET elegiré C# (Visual C#, adaptación para Microsoft).

C# es un lenguaje de programación que se ha diseñado para compilar diversas aplicaciones que se ejecutan en .NET Framework. C# es simple, eficaz, con seguridad de tipos y orientado a objetos.

Las numerosas innovaciones de C# permiten desarrollar aplicaciones rápidamente y mantener la expresividad y elegancia de los lenguajes de estilo de C.
Visual C# es una implementación del lenguaje de C# de Microsoft. Visual Studio ofrece compatibilidad con Visual C# con un completo editor de código, un compilador, plantillas de proyecto, diseñadores, asistentes para código, un depurador eficaz y de fácil uso y otras herramientas. La biblioteca de clases de .NET Framework ofrece acceso a numerosos servicios de sistema operativo y a otras clases útiles y adecuadamente diseñadas que aceleran el ciclo de desarrollo de manera significativa.

Las principales especificaciones y características del framework .NET las resumo aquí:

- Proporcionar un entorno coherente de programación orientada a objetos, en el que el código de los objetos se pueda almacenar y ejecutar de forma local, ejecutar de forma local pero distribuida en Internet o ejecutar de forma remota.

- Proporcionar un entorno de ejecución de código que minimiza los conflictos en el despliegue y versionado de software.

- Ofrecer un entorno de ejecución de código que promueva la ejecución segura del mismo, incluso del creado por terceras personas desconocidas o que no son de plena confianza.

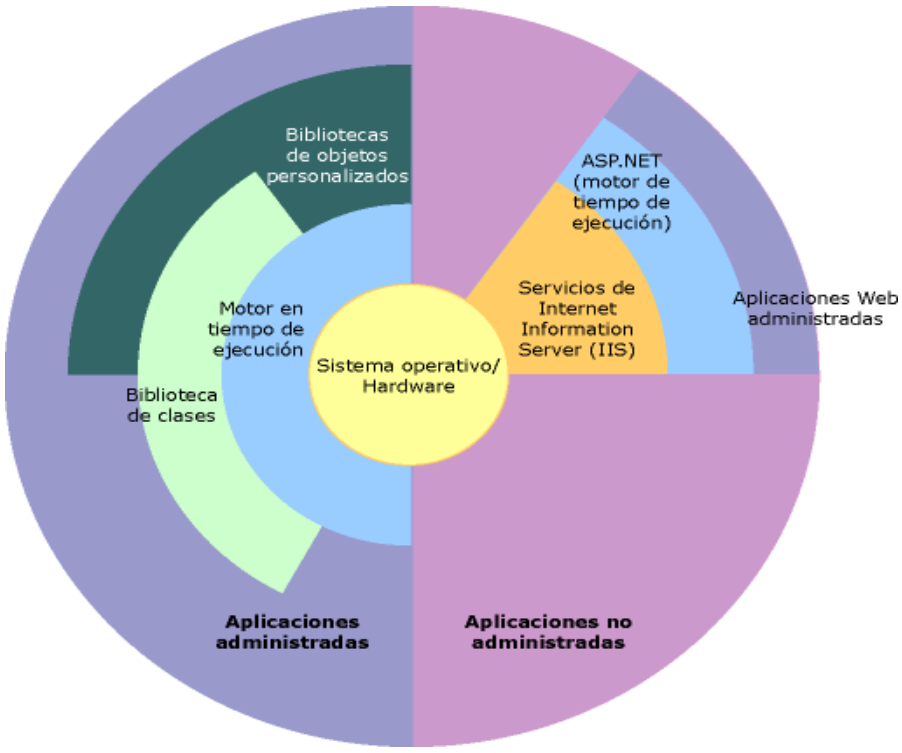
- Proporcionar un entorno de ejecución de código que elimine los problemas de rendimiento de los entornos en los que se utilizan scripts o intérpretes de comandos.
- Ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en Windows o en el Web.

- Basar toda la comunicación en estándares del sector para asegurar que el código de .NET Framework se puede integrar con otros tipos de código.

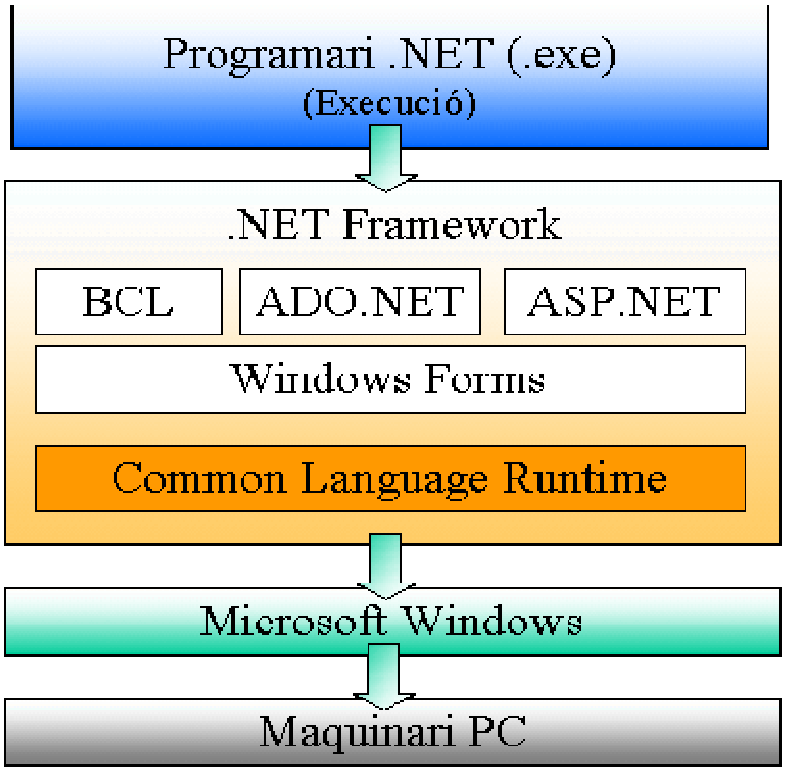
.NET Framework consta de dos componentes principales: Common Language Runtime y la biblioteca de clases de .NET Framework. Common Language Runtime es el fundamento de .NET Framework.

El motor en tiempo de ejecución se puede considerar como un agente que administra el código en tiempo de ejecución y proporciona servicios centrales, como la administración de memoria, la administración de subprocesos y la comunicación remota, al tiempo que aplica una seguridad estricta a los tipos y otras formas de especificación del código que promueven su seguridad y solidez. De hecho, el concepto de administración de código es un principio fundamental del motor en tiempo de ejecución.

El código destinado al motor en tiempo de ejecución se denomina código administrado, a diferencia del resto de código, que se conoce como código no administrado.La biblioteca de clases es una completa colección orientada a objetos de tipos reutilizables que se pueden emplear para desarrollar aplicaciones que abarcan desde las tradicionales herramientas de interfaz gráfica de usuario (GUI) o de línea de comandos hasta las aplicaciones basadas en las innovaciones más recientes proporcionadas por ASP.NET, como Web Forms y Servicios Web XML.
Distribución sistemas y aplicaciones .NET:



A continuación pondré un esquema multicapa de como actúa esta tecnología en el sistema operativo:



Ahora voy a exponer brevemente las características y fundamentos principales del lenguaje C# que es el utilizado en mi caso con .NET para el desarrollo del software controlador. Documentación obtenida de la página web oficial de microsoft [6].

C# es un lenguaje orientado a objetos elegante y con seguridad de tipos que permite a los desarrolladores compilar diversas aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Puede utilizar C# para crear aplicaciones cliente de Windows, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y mucho, mucho más. Visual C# proporciona un editor de código avanzado, cómodos diseñadores de interfaz de usuario, depurador integrado y numerosas herramientas más para facilitar el desarrollo de aplicaciones basadas el lenguaje C# y .NET Framework.

La sintaxis de C# es muy expresiva, pero también es sencilla y fácil de aprender. La sintaxis de C# basada en signos de llave podrá ser reconocida inmediatamente por cualquier persona familiarizada con C, C++ o Java.

Los desarrolladores que conocen cualquiera de estos lenguajes pueden empezar a trabajar de forma productiva en C# en un plazo muy breve. La sintaxis de C# simplifica muchas de las complejidades de C++ y proporciona características eficaces tales como tipos de valor que admiten valores NULL, enumeraciones, delegados, expresiones lambda y acceso directo a memoria, que no se encuentran en Java.

C# admite métodos y tipos genéricos, que proporcionan mayor rendimiento y seguridad de tipos, e iteradores, que permiten a los implementadores de clases de colección definir comportamientos de iteración personalizados que el código cliente puede utilizar fácilmente. Las expresiones *Language-Integrated Query* (LINQ) convierten la consulta fuertemente tipada en una construcción de lenguaje de primera clase.

Como lenguaje orientado a objetos, C# admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método Main que es el punto de entrada de la aplicación, se encapsulan dentro de definiciones de clase. Una clase puede heredar directamente de una clase primaria, pero puede implementar cualquier número de interfaces.

Los métodos que reemplazan a los métodos virtuales en una clase primaria requieren la palabra clave **override** como medio para evitar redefiniciones accidentales. En C#, una struct es como una clase sencilla; es un tipo asignado en la pila que puede implementar interfaces pero que no admite la herencia.

Además de estos principios básicos orientados a objetos, C# facilita el desarrollo de componentes de software a través de varias construcciones de lenguaje innovadoras, entre las que se incluyen las siguientes:

- Firmas de métodos encapsulados denominadas *delegados*, que habilitan notificaciones de eventos con seguridad de tipos.

- Propiedades, que actúan como descriptores de acceso para variables miembro privadas.

- Atributos, que proporcionan metadatos declarativos sobre tipos en tiempo de ejecución.

- Comentarios en línea de documentación XML.

- Language-Integrated Query (LINQ) que proporciona funciones de consulta integradas en una gran variedad de orígenes de datos.

Si necesita interactuar con otro software de Windows, como objetos COM o archivos DLL nativos de Win32, podrá hacerlo en C# mediante un proceso denominado "interoperabilidad". La interoperabilidad habilita los programas de C# para que puedan realizar prácticamente las mismas tareas que una aplicación C++ nativa. C# admite incluso el uso de punteros y el concepto de código "no seguro" en los casos en que el acceso directo a la memoria es totalmente crítico.

El proceso de compilación de C# es simple en comparación con el de C y C++, y es más flexible que en Java. No hay archivos de encabezado independientes, ni se requiere que los métodos y los tipos se declaren en un orden determinado. Un archivo de código fuente de C# puede definir cualquier número de clases, structs, interfaces y eventos.

En el tema de la subida de ficheros al servidor web opté por una política *first in first out* de manera inmediata. Es decir, cuando se genera un recurso (imagen, grabación, fichero …) lo subo inmediatamente al servidor, para no generar una cola que más adelante pudiera suponer problemas de congestión.

Compiladores y Procesadores del lenguaje

Documentación de compiladores y procesadores del lenguaje [7]. Se trata de un campo bastante extenso donde se tratará de explicar un poco las nociones básicas sobre compiladores y procesadores del lenguaje junto con los tipos que hay hoy en día.

Según la Real Academia de la Lengua Española, compilar consiste en reunir en un solo texto extractos o fragmentos de obras ya publicadas. Sin embargo, en informática el término compilar se emplea con un significado similar a traducir. Podemos ver un traductor como algo que recibe un conjunto de datos que representan un mensaje en un lenguaje determinado y genera un nuevo conjunto de datos que representan el mismo mensaje en otro lenguaje. Centrándonos en lenguajes de programación, los datos de entrada son una expresión de un programa en un determinado lenguaje, denominado lenguaje fuente, y los de salida la expresión del mismo programa en un lenguaje diferente, el lenguaje objeto. Un programa expresado en lenguaje fuente suele denominarse programa fuente mientras que el programa obtenido tras su traducción se denomina programa objeto. El proceso de traducción modifica la expresión del programa pero no su semántica (el conjunto de órdenes que el programa expresa). Si el lenguaje fuente es un lenguaje de alto nivel y el lenguaje objeto es código máquina (normalmente reubicable) o ensamblador, el traductor se denomina compilador.

Dicho esto toda esta teoría es aplicable para identificar ciertos patrones de texto mediante un procesador del lenguaje de manera que si se reconoce un cierto patrón adecuado a dicho procesador haga una determinada acción (en el caso de un compilador sería traducir a lenguaje objeto).

Todo procesador de lenguaje contiene un analizador sintáctico (*parser*) que permite determinar si una entrada se ajusta a la gramática o no. Para ello, lo primero que necesita el parser es recibir los tokens de la entrada uno a uno a través de un analizador léxico.

El analizador léxico consiste en un programa que va leyendo uno a uno cada *token* (unidad mínima de un universo) de la entrada y se lo pasa al parser.

Un parser es un programa que toma como entrada una cadena de unidades sintácticas y comprueba si pertenece a una gramática. La salida de un parser puede tener diferentes formas. En teoría el parser debe devolver un árbol de análisis sintáctico que muestre la estructura sintáctica de la cadena analizada. Sin embargo, es común que las fases de análisis sintáctico y generación de código intermedio de un compilador vayan unidas (traducción dirigida por la sintaxis) de modo que la salida del parser sea ya una representación en código intermedio.

Existen diversas metodologías para construir parsers, aunque se basan en dos técnicas básicas. En realidad se trata de dos formas de ir construyendo el árbol de análisis sintáctico: En los métodos denominados *bottom-up* se comienza construyendo el árbol a partir de las hojas mientras que en los llamados *top-down* se parte de la raíz hasta llegar a las hojas.

La técnica *bottom-up* más habitual es la llamada *shift-reduce* en la que el parser determina en cada paso si ha leído la parte derecha de una producción y, en tal caso, la cambia por la parte izquierda de la producción leída. Podemos construir diferentes tipos de parsers *shift-reduce*. Un ejemplo de esta técnica son los parsers de precedencia operador (limitados) y parsers LR(k) (mucho más potentes).

Las técnicas *top-down* crean el árbol sintáctico empleando métodos recursivos descendentes. Uno de los parsers top-down más habituales es el llamado parser predictivo que trata de adivinar, en función del token de entrada, cuál de las producciones de un no terminal debe aplicarse. Los parser *top-down*, en general, suelen dar mejores resultados a la hora de resolver errores. Los parsers predictivos LL son un ejemplo de parsers top-down.

La solución para la implementación del parser sobre terrorismo integrado en el software controlador fue una de las más complicadas y difíciles de encontrar del proyecto.

En un primer momento pensé en desarrollarlo todo yo mismo en C# para evitarme posibles problemas posteriores de integración. Pero en cambio tenía la complejidad clara de lo que supone desarrollar un parser LALR. Empezé por las estructuras de datos como pila, tabla de símbolos … pero me di cuenta que era un sobreesfuerzo bastante grande.
Sobre todo teniendo en cuenta que existen herramientas, bastante potentes, que a partir de la especificación de una gramática te generan un parser totalmente funcional.
Ya había tenido alguna experiencia sobre alguna de estas herramientas. Concretamente para sistemas UNIX, las herramientas bison y flex te generan un parser en lenguaje C a partir de una gramática y expresiones regurales. Pero para este proyecto necesitaba una herramienta para el sistema operativo Windows. Y el parser debía ser producido en C#.

Dicho esto, la solución la encontré con la herramienta GOLD Parser Builder.

A continuación expondré las características más relevantes de la herramienta. Esta documentación ha sido obtenida de la documentación de la página oficial de GOLD Parser Builder. [8]

Overview

The GOLD Parsing System is designed to aid in the development of compilers, interpreters and translators while supporting multiple programming languages. To accomplish this goal, the system takes advantage of the LALR and DFA algorithms. Both of these can be implemented with simple state transition graphs. As a result, it is very easy to write the algorithms in different programming language since the logic simply looks up a value in a table and then acts accordingly. The creation of these tables is where all the "thinking" takes place.
The system is composed of two different components, the "Builder" and the "Engine", and the a file that is stores table information. The "Builder" constructs the parse tables. The "Engine" uses the tables.

The Components

The Builder

The Builder is used to read a source grammar written in the GOLD Meta-Language, produce the LALR and DFA parse tables, and, finally, save this information to Compiled Grammar Table file.

The Builder Application runs on the Windows 32-bit operating systems which include, but are not limited to, Windows 9x, Windows NT and Windows XP. The application also contains a number of features which could have been implemented as different programs. Each feature was included to create an easy-to-use integrated development environment. These include the ability to create skeleton programs and the ability to interactively test a grammar.

There is also a command-line version of the Builder. In the future, the Builder may be ported to different operating systems such as UNIX, Linux, and Mac-OS.

Compiled Grammar Table File

The Compiled Grammar Table file is used to store table information generated by the Builder.

The Engine

The Engine component performs the actual parsing. The Engine can be developed in any programming language necessary. Since all the complex work was already performed the Builder, the Engine simply needs to read the Compiled Grammar Table file and implement the LALR and DFA automatas.

As different implementations of the Engine are created for different programming languages, the approach and design will vary. Since different programming languages use different approaches to designing programs, each implementation of the Engine will be a little bit different. As a result, an implementation of the Engine written for Visual Basic 6 will differ greatly from one written for ANSI C.

Due to the work of kind and generous contributors, there are numerous Engines at your disposal. Please visit the Engine Download page for some more information.

Development Overview

Step 1.

The first step to designing your compiler or interpreter is to write a grammar for the language being implemented. The description of the grammar is written using any text editor - such as Notepad or the editor that is built into the GOLD Builder. This does not require any coding.

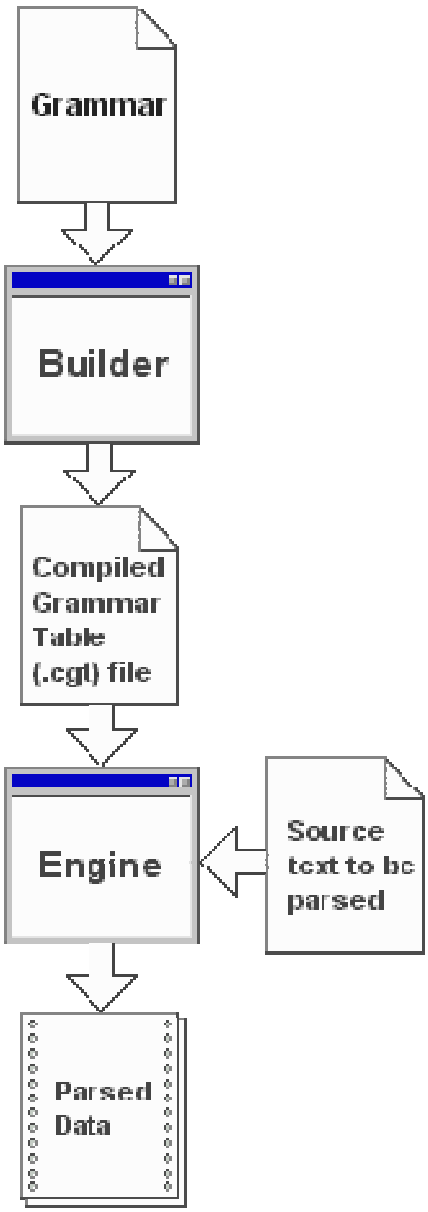
Step 2.

Once the grammar is complete, it is analyzed by the GOLD Builder. During this process, LALR and DFA parse tables are constructed and any ambiguities or problems with the grammar are reported. Once the grammar is analyzed, the tables are saved to a Compiled Grammar Table file to be used later by the actual parsing engine. At this point, the GOLD Parser Builder is no longer needed - having performed its duties.

Step 3.

Finally, the parse tables are read by the parsing engine. The Engine can be implemented in any number of programming languages. Each Engine works a little differently - using the programming style and paradigm that best suites the language.
In all cases, the source text is analyzed by the parser engine and a parse tree is constructed.

Esquema genérico sobre la herramienta GOLD Parser Builder



Expuesto el funcionamiento de la herramienta debo añadir, que para mi proyecto he usado las librerías GoldParserEngine.dll como Engine y CalithaLib.dll para la generación del parser en C#.

Sobre la integración del software controlador con el sistema, opte por la realización de una serie de scripts en PHP, de manera que el software controlador actuase como una especie de cliente que se comunicaba e interactuaba con el servidor web por medio de estos scripts. Desde la consulta de peticiones, modificación de parámetros, subida de recursos al servidor, captación de datos del servidor, envío de notificaciones … se encargan estos scripts.
Otro tema que tuve que abordar fue cómo generar los recursos que se le piden al Software Controlador y de qué manera para poder utilizarlos de la forma más estándar posible.

Teniendo esto en cuenta necesitaba que el programa generara imágenes con formato JPG (formato de compresión de imagen estándar), archivos de sonido WAV (fácilmente reproducidos por cualquier reproductor de sonido estándar) y ficheros PDF (mundialmente conocido, formato ligero, legible por todos los lectores PDF existentes).

De esta manera me acercaba mucho al objetivo final de hacer un ejecutable estándar, funcional y operativo para todas las versiones de sistemas operativos Windows y navegadores (a la hora de mostrar los recursos al usuario).

Pero dicho todo lo anterior, me faltaba lo verdaderamente importante: crear/importar una serie de librerías que me permitieran generar todos esos recursos.
Entonces buscando por Internet y después de encontrar y evaluar distintas alternativas para cada uno de ellos me decidí por:

- AForge.Video: para el control y manejo de la webcam para Sistemas Operativos Windows. Obviamente también me permitía elegir el formato de salida requerido.

- NAudio: para el control del micrófono del ordenador monitorizado. Totalmente compatible con Windows, permite el salvado del archivo de audio en el formato requerido.

- iTextSharp: librería estándar que permite la generación de documentos con formato PDF, en este caso a través de la tecnología .NET C#.

Todas estas librerías son nativas para Windows (dll) lo que hace que su funcionamiento y tiempo de ejecución sean altamente eficaces y eficientes.

En cuanto a los posibles errores o incidencias que se pudiesen dar durante el funcionamiento del software controlador la solución consiste básicamente, en programar bien. Parece tópico pero en realidad es así. Manejar muy bien y de manera eficiente el manejo de excepciones, procesos, conexiones, serializaciones...

Todas estas medidas son totalmente necesarias en fallos de conexión de red, retardos, congestiones... Se deben desarrollar bien para que el programa pueda seguir operativo.
Y por supuesto hay que tener en cuenta los errores de tipo humano. Para ello, validar campos, notificaciones... son necesarios para minimizarlos.

Y hablando concretamente de la programación en .NET tener especial cuidado con la conexión entre clases, sobre todo en la utilización de delegados en el paso de objetos de clases GUI.

C# es un lenguaje muy parecido a Java en cuanto a sintaxis (obviamente hay cosas que son distintas), por eso se debe aprovechar la "facilidad" de escritura que tiene, pero teniendo en cuenta todos los errores anteriormente descritos que pueden suceder

Por último llegamos al problema de seguridad más grande que tuve durante el desarrollo del Software Controlador, la autenticidad del usuario.

Como ya he explicado, la idea original, y a mi entender la correcta es que uno y solo un usuario correctamente registrado y a más adelante logueado pudiese tener un acceso correcto al programa.
Esto quiere decir que un par de credenciales, usuario y contraseña no válidos produzca un error de autentificación y el software no permita el acceso.

Hasta este punto, todo correcto. Trivial más bien. Cualquier sistema de autenticación debería como mínimo garantizar eso. Pero en mi caso debía ir un poco más allá.
Quería garantizar la exclusividad de un usuario y contraseña validos para un único terminal. Es decir, que ningún otro para de usuario y contraseña válidos pudiese generar un acceso correcto en un mismo terminal.
Entonces empecé a pensar en posibles soluciones que podía llevar a cabo.

La primera de ellas fue la de generar un código aleatorio ÚNICO en el momento del registro del usuario y guardarlo en un fichero, que sirviera como identificador exclusivo del usuario y contraseña justamente introducidos.
Funcionalmente era una solución, en mi opinión bastante buena. Pero tenía un problema de seguridad física, ya que encontrando dicho fichero en el sistema de ficheros del software y accediendo a el podrías averiguar de cual era dicho código y modificarlo por otro, que daría acceso correcto para otro para de usuario y contraseña distinto.

Entonces se me ocurrió como solución para no suplantar dicho código el guardarlo en el fichero pero codificado con un algoritmo de encriptación con base matemática. En mi caso el algoritmo asimétrico de clave pública y clave privada RSA. De esta manera si alguien ajeno tenía acceso al fichero donde estaba el código lo vería encriptado y obviamente no lo podría sustituir por otro (distintas claves). No tendría acceso con otro usuario y contraseña válidos.
Pero este método también tenía un gran fallo. Efectivamente no se puede suplantar la identidad pero si se puede provocar un fallo de denegación de servicio para el usuario legítimo. Simplemente, modificando el código encriptado del fichero o borrando dicho fichero el usuario autorizado ya no podría alojarse en el sistema.

Llegue a la conclusión de que lo que necesitaba era un identificador único (para asociarlo a las credenciales) pero que no estuviese reflejado "físicamente" ni en un fichero, ni una base de datos... Necesitaba tener acceso a él siempre que quisiese (en la autentificación del usuario) y que me garantizase la unicidad del mismo.
Pensando en todo ello me decanté por la dirección física de la tarjeta de red del terminal, la dirección MAC. Desarrollé una función que en tiempo de ejecución obtenía dicha dirección física como identificador único. De esta manera me quitaba todos los problemas de las otras dos alternativas.

Es así como logré la unicidad y la exclusividad para uno y solo un par de usuario y contraseña válidas en cada terminal.

Tengo que decir, que siempre hay técnicas y métodos por los cuales alguien puede encontrar vulnerabilidades para obtener acceso no autorizado a un sistema. Pero obviamente ya no es tan trivial como lo era en los dos primeros casos.

A continuación voy a detallar las soluciones adoptadas para los problemas más importantes que me surgieron en el desarrollo de la página web.

En el tema del diseño, de la creación de interfaces web, es lógico pensar que se debe encontrar unos estilos que se adapten a todo tipo de pantallas. Dichas pantallas pueden diferenciarse en tamaño, resolución... Por eso, es imprescindible decidir una buena política de estilos que permitan dotar a la web de un diseño atractivo y totalmente *responsive* (adaptativo) para cada uno de los diferentes tipos de dispositivos existentes.
En un primer momento decidí hacer el html y los estilos con css propios, es decir, diseñando y desarrollando yo todo. Pero la verdad, es que el proyecto en sí ya es bastante largo, y opté por utilizar la librería BootsTrap que exporta Twitter.

A continuación voy a exponer la historia, características y propiedades fundamentales de Bootstrap.

Twitter Bootstrap es una librería de codigo abierto para el desarrollo de interfaces web. Consiste en una serie de módulos css3 y JavaScript. Ambas partes son en enlazadas con códido de marcas HTML5.

Se puede crear todo tipo de estructura gracias a estos estilos, a la vez que permite muchas animaciones y efectos debido a funciones de JavaScript.

Es una librería que permite la creación de interfaces totalmente responsive, es decir, que se adapta al tamaño y resolución según el tipo de dispositivo con que se visualiza, ya sea ordenadores, tablets o smartphones.

Es casi totalmente compatible con cualquier navegador. Actualmente se encuentra en la versión 3.

Dentro de la web, en la zona de imágenes resultantes por la webcam debía desarrollar un sistema de galerías atractivo y funcional que permitiese al usuario la visualización de las mismas acorde a la especificación detallada del proyecto.
Es un caso similar al anterior. Tenía básicamente dos opciones, diseñar y desarrollar toda la galería o importar algunas librerías, css para los estilos y javascript para la animación.

Por otro lado hay ciertas funcionalidades que se quiere dotar a la web que requieren la presencia de alguna librería externa que facilite y optimice el funcionamiento. Es el caso para cuando se quiere enviar un correo electrónico desde el sistema.

La tecnología de servidor que yo utilizo en la web es PHP. Busque por tanto una librería para este lenguaje. Me decidí por PHPMailer. Bastante funcional y operativa. Tiene muy bien implementados todos los protocolos de seguridad necesarios.
De esta forma cuando un usuario quiere compartir un recurso vía email la solución adoptada es bastante óptima.

A la hora de estructurar la disposición de contenidos dentro de los interfaces de la web hay que tener en cuenta que debe quedar bastante limpio, claro e intuitivo. Por eso lo que me decidí antes de diseñar todo finalmente fue que cada dispositivo tenía que tener claramente diferenciada su zona. Por ello lo más fácil es disponer de un menú principal con acceso rápido a cada zona.

Dentro de cada zona será necesario un submenú para tener acceso a todas las opciones que la web debe ofrecer al usuario, bandeja de entrada para mostrar los recursos al usuario, un espacio para el envío y cancelación de una nueva petición al Software Controlador y una zona para la modificación de los parámetros de monitorización del dispositivo en cuestión.

A continuación voy a relatar las soluciones que adopté para los problemas del Remote Control Software.

De la parte del servidor una de las primeras cosas en las que quise innovar fue en el control de acceso. En lugar de diseñar el típico método de autenticación mediante usuario y contraseña me dispuse a intentar desarrollar un algoritmo de detección y reconocimiento facial que validase o no un posterior acceso al software.

El software del server está desarrollado en Java. Por tanto entre las dos herramientas más potentes que conozco de visión por ordenador MATLAB y OpenCV me decanté por la segunda para desarrollar el sistema de autentificación facial.
Antes de exponer las principales características de OpenCV, relataré a grosso modo cual fue la técnica que empleé.

Desarrollé una beta a partir del cual con una variedad de imágenes de la misma cara con diferencies matices y factores (posiciones de la cara, luminosidad, con o sin barba...) calculé mediante el algoritmo del PCA una serie de eigenvalues y eigenvectores. A esto se le llama entrenamiento.
Después del entrenamiento fui a probar el sistema con una imagen que se correspondía con la cara del entrenamiento realizado y con otra cara distinta.
Obviamente con un umbral de error (era una beta, harían falta muchas más imágenes para el entrenamiento) elevado y por optimizar el sistema ya era capaz de proporcionar acceso a la cara correcta y denegarlo a la otra.

A continuación voy a explicar y detallar lo principal de esta librería de visión artificial llamada OpenCV. Documentación oficial OpenCV [9].

OpenCV es una librería, con licencia de código abierto que contiene una serie de recursos para la elaboración de algortimos de visión por computador.

Originalmente diseñada para C++ pero con versiones y wrappers ya desarrollados para Java, Python... Con una capacidad de procesamiento de imagen de entre 25 y 30 frames por segundo. Compatible con cualquier IDE utilizado hoy en día como Eclipse, Netbeans...
Posee una estructura modular donde cada módulo contiene un determinado tipo de clases: *core, imgproc, videl, calib3d, feature2d, objdetect, highgui, gpu*.

Otra de las cosas, que en la especificación del Remote Control Software quise dejar

claro fué que el server puede en cualquier momento cerrar la conexión y no tener que quedar a merced de la voluntad del cliente.

Pero dicho esto, se debe cerrar todo bien de manera que no salte ninguna excepción. Por eso añadí una ramificación más en el protocolo de conexión que diseñe entre cliente y server del software.

Después de diseñar e implementar la función principal de este software, la de manejar completamente un ordenador desde otro con interfaz gráfica me decidí a incorporar una herramienta de transmisión de ficheros en ambas direcciones, pero que obviamente, no podía interrumpir la conexión ya establecida.
Por lo cual lo que hice fue la apertura de una segunda conexión en paralelo en otro puerto (bajo el mismo protocolo TCP) de manera que las transmisiones de los bytes de los ficheros no afectaban en absoluto a la función principal del software.

En la parte del cliente se debe poder ofrecer la posibilidad de generar un informe con los datos de la conexión, siempre y cuando haya sido finalizada previamente (por parte del cliente o por parte del servidor). Una de las estadísticas que se muestran en el informe será representada en un gráfico.

Por lo tanto lo que hice fue decidirme por un par de librerías externas que me permitieran llevar a cabo dicha especificación.

Me decidí por JFreeChart para la realización de gráficos. Dicha librería permite a través de la utilización de unas clases con sus métodos customizados con los parámetros en cuestión generar una imagen con el tipo de gráfico que se desee, líneas, barras, sectores...
Por otra parte, para la generación de un documento PDF opté por la librería open source iText.
Permite a través de la creación y llamada a unos métodos de una clase la renderización de un documento PDF con bastante variedad en lo que a formato se refiere, tipos y tamaños de letra, colores, tabuladores, inserción de imágenes...

En el cliente también es obligatorio que la función principal y la transmisión o recepción de ficheros sea concurrente. Pero veo redundante volver a explicar la solución a este problema ya que forma parte de la misma solución explicada en el server (dos conexiones).

En cuanto al tema de cierre de conexión en el cliente opté por proporcionar al usuario diferentes modos según su conveniencia en cada momento.
Para ello realicé un protocolo de desconexión con distintas variantes.
La primera de ellas era la de cerrar conexión con el server pero manteniendo el server operativo para una futura conexión.

La segunda es la de cerrar la conexión, mantener el server operativo y cerrar el software cliente. Se le ahorra al cliente el cerrar el programa.
La última es cuando quieres cerrar todo el sistema de manera que se cancela correctamente la conexión, se manda cerrar al servidor (deja de estar operativo) y por último se cierra automáticamente el cliente.

En esta última parte del apartado *soluciones a problemas planteados* detallaré aquellas que tuve que tomar para la Android app.

En cuanto al interfaz, al igual que en la web debía quedar bastante claro las zonas de cada uno de los dispositivos y las opciones en cada uno de ellos.

Lo que opté por hacer es la utilización de un menu principal en el que había un item para cada uno de ellos. La presión sobre cada uno de ellos reflejaba en la pantalla una disposición jerarquica de layouts de manera que en la parte superior se reflejaba un submenu con las opciones bandeja de entrada, mandar/cancelar una nueva petición o modificación de parámetros.

Inmediatamente después se ve reflejado ya en el resto de la pantalla el contenido de cada item de este submenú.
De esta forma consideré que quedaba bastante claro el mostrado del contenido y fácil la navegación por las distintas opciones de la aplicación.

Otro de los "problemas" que tuve fue el mecanismo a desarrollar para una inmediata autentificación del usuario después de haber introducido usuario y contraseña la primera vez o la primera vez después de haber elegido *desconexión* .
Se me ocurrió utilizar las preferencias del sistema operativo Android (SharedPreferences) por la cual después de las primeras conexiones guardaba un identificador.
Entonces el funcionamiento era muy simple. Si existía un parámetro a modo de identificador en las preferencias no era necesario pedir autentificación y se accedía directamente a la aplicación.

Cada vez que el usuario elegía *desconexión* del menú principal de la aplicación se eliminaba el parámetro de las preferencias. De esta manera, en un futuro acceso a la aplicación, ésta te pedía las credenciales de usuario y contraseña para poder volver a entrar.

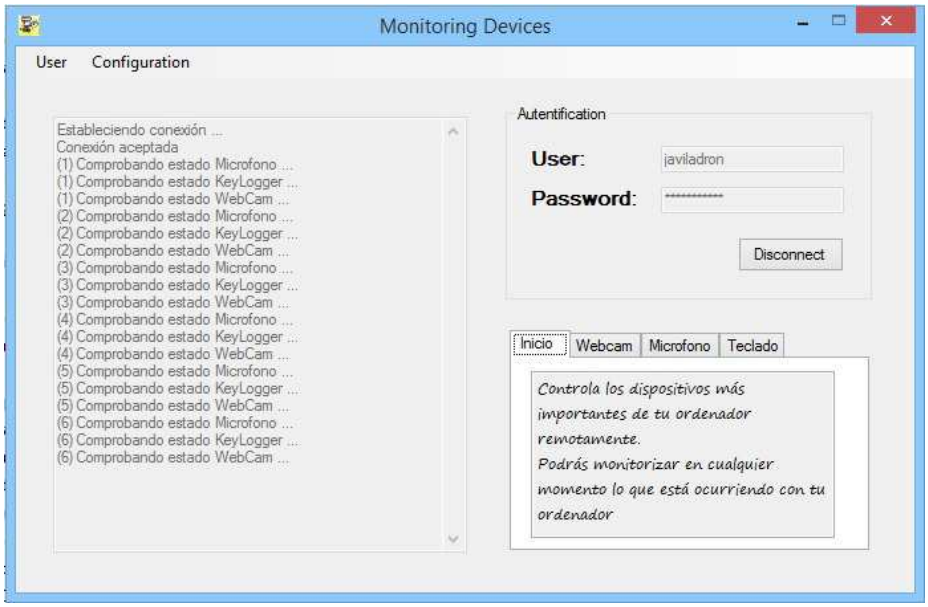
Por último decir que me aseguraba siempre, en cualquier paso de la aplicación si existía conexión a Internet porque es necesario estar en constante comunicación con el servidor del sistema para el correcto funcionamiento de la aplicación.

4.1 Pruebas unitarias

En este apartado reflejaré las pruebas que hice para comprobar que todo el sistema funciona correctamente. En este caso concreto son pruebas unitarias. Más adelante me mostraré pruebas de carga para evaluar tiempos, concurrencia...

Con las pruebas unitarias voy a seguir el mismo esquema por el que optado en varios apartados de esta memoria. Iré desglosando por apartados según las grandes partes de las que consta el sistema.

Pruebas unitarias para el software controlador.



- La primera y una de las más importantes era la autentificación según el sistema ya explicado.

Por eso para mi ordenador personal, por ejemplo, me cercioré de que el software solo me proporciona acceso a mí con mi usuario y contraseña y no admite ningún otro para de usuario y contraseña válidos.

- La función principal del software es la de monitorizar parámetros de los dispositivos, generar los recursos pertinentes en caso de solicitud y subirlos correctamente al servidor web. Por tanto realicé pruebas individuales para cada uno de estos puntos.

1) Pruebas de consulta con el servidor web para la consulta de parámetros de los dispositivos. Para ello se desarrolla un cliente HTTP que realiza una petición del protocolo con el servidor web y le envía los datos serializados en un string. Prueba correcta.

2) Generación de recursos. Las librerías, iTextSharp, AForge.Video y NAudio, antes mencionadas, generan cada uno de los recursos correspondientes de manera correcta. Prueba correcta.

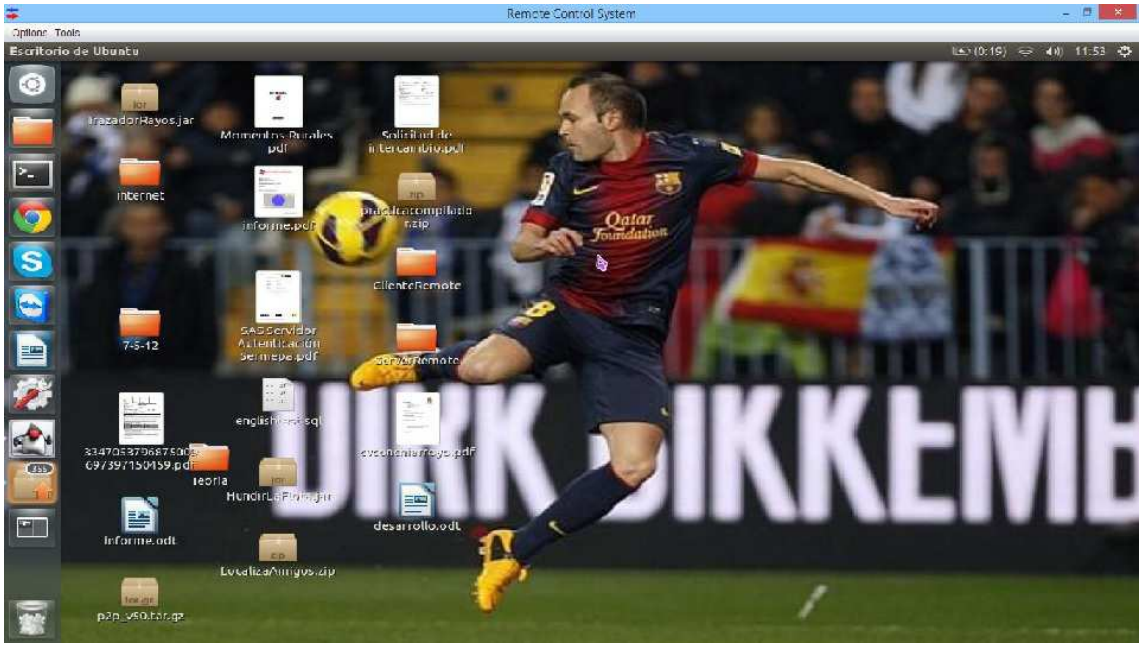
3) Subida de recursos. Cliente http que realiza una conexión con el servidor para la subida correcta de cada uno de los distintos recursos. Prueba correcta.

- Función del parser integrado en el software controlador generado a partir de la herramienta GOLD Parser Builder.
Pruebas en la propia herramienta una vez diseñada la gramática. Dichas pruebas son cadenas que deben validarse y otras que no a partir de la gramática especificada. Hay un ejemplo claro en el apartado de análisis y diseño de como actúa el parser para una cadena válida.
Por otro lado, una vez generado el parser con las librerías correspondientes para el lenguaje C# pruebas unitarias de aceptación de cadenas validas con su correspondiente envío de una alarma al servidor web a traves de una peticion cliente http.
Pruebas superadas correctamente.

- Pruebas unitarias para cada una de las acciones llevadas a cabo por el usuario manualmente que tienen una interacción con el servidor web del sistema. En estas pruebas queda excluida la autentificación ya testada anteriormente. Dichas pruebas son para la creación y eliminación de usuario y para la modificación de parámetros en de los dispositivos.
Tratada cada una individualmente funcionan correctamente. Pruebas superadas.

A continuación voy a exponer las pruebas unitarias realizadas para el software de control remoto.

En la parte del cliente.

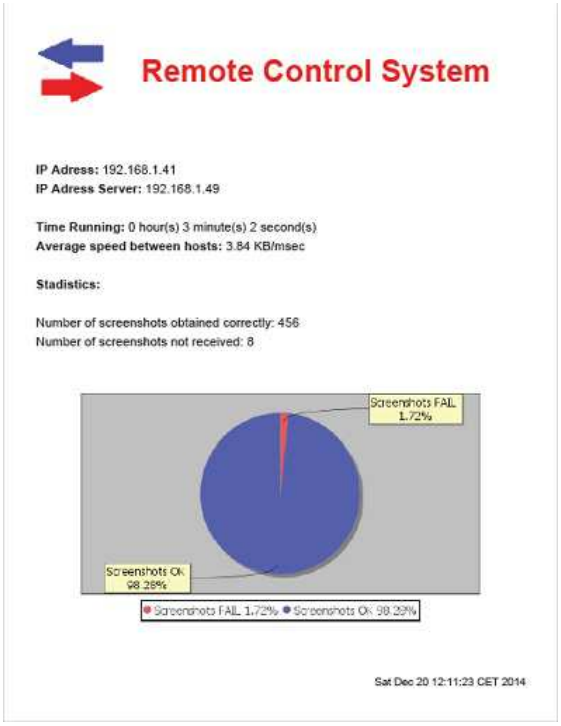


- Correcta captura de eventos de teclado y ratón. Absolutamente clave. Posición del puntero correcta y teclas pulsadas coincidentes. Prueba superada.

- Envío correcto de los eventos por la conexión principal. Pruebas superadas.

- Recepción y transformación posterior en imagen de los bytes recibidos del server por la conexión principal de los pantallazos capturados. Prueba superada. Una vez correcto este paso representación visual en el cliente de dicha imagen para mostrarla al usuario. Prueba superada.

- Renderización del informe de la conexión en PDF. Muy testada para garantizar datos correctos y coherentes.



En la parte del servidor.

- Correcta autentificación por reconocimiento facial. En el apartado de desarrollo en la sección de soluciones adoptadas muestro unas pruebas del entrenamiento realizado. Prueba superada.

- Correcta captura de imágenes de la pantalla, posterior transformación y envío por la conexión principal. Prueba superada.

- Recepción de los eventos de teclado y ratón de la conexión principal enviados por el cliente. Prueba superada.

- Correcta ejecución de dichos eventos que es lo que permite el control remoto desde la otra parte. Absolutamente clave que funcione correctamente. Prueba superada.

Para ambas partes. Todas ellas superadas correctamente.

- Control y manejos de excepciones en la creación y destrucción de conexiones entre ambas partes.

- Envío correcto de parámetros, clases, estructuras de datos... a través de la conexión principal para garantizar el buen funcionamiento de los protocolos diseñados y opciones del software ofrecidos al usuario.

- Envío correcto de ficheros a través de la conexión auxiliar para garantizar que la herramienta FTP incorporada funciona correctamente.

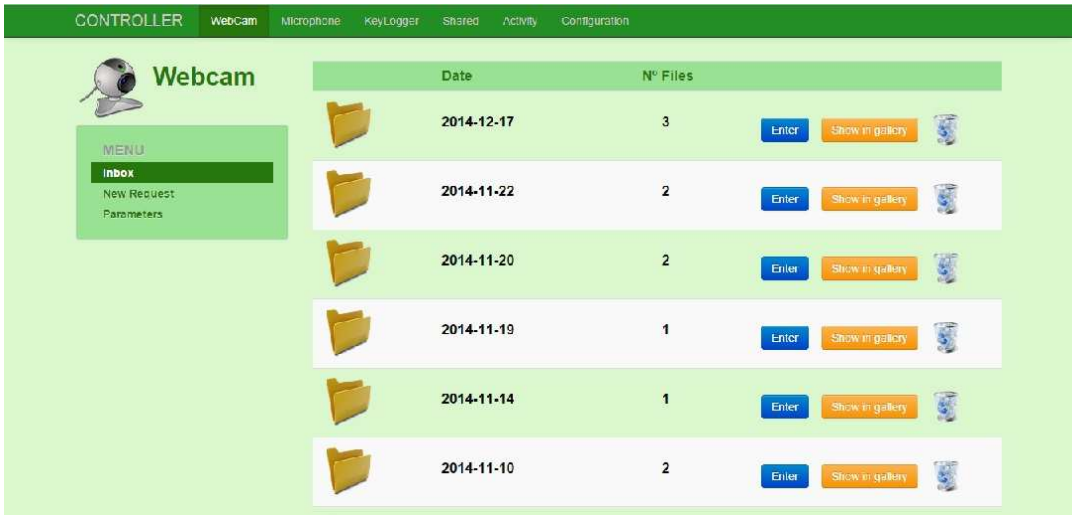
Pruebas unitarias para la página web.

- Pruebas en la autentificación del usuario. Debe aceptar los pares válidos y rechazar los falsos. Prueba superada.

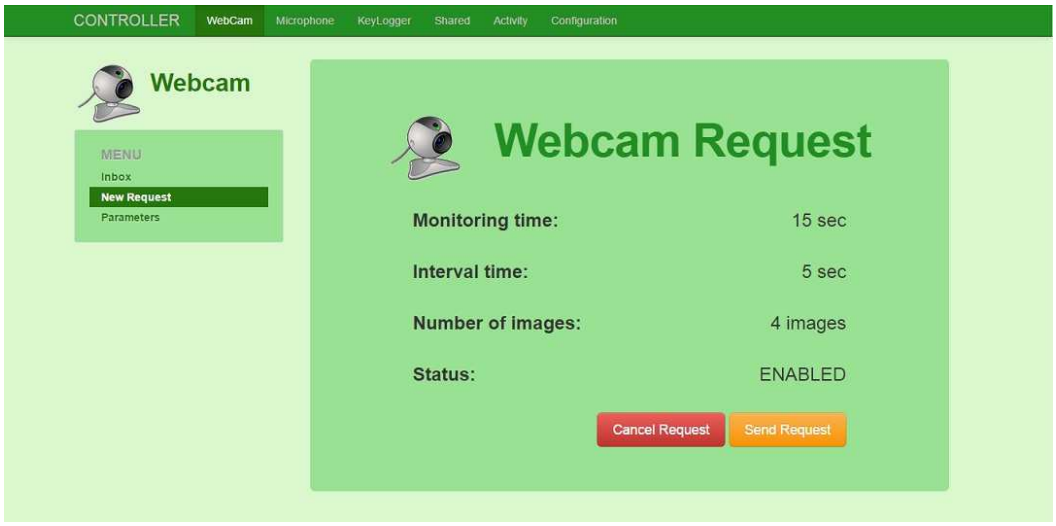
- Pruebas de SQL injection para garantizar la no suplantación por consultas envenenadas a la base de datos. Escape de caracteres no válidos. Prueba superada.

- Correcta descarga de los softwares en la web. A saber, Software Controlador, Remote Control y Android app. Prueba superada.

- Correcta organización de los recursos de cada uno de los dispositivos en las bandeja de entrada acorde a la especificación del sistema. Consiste en asegurarse de que la consulta es correcta a la base de datos. Prueba superada.

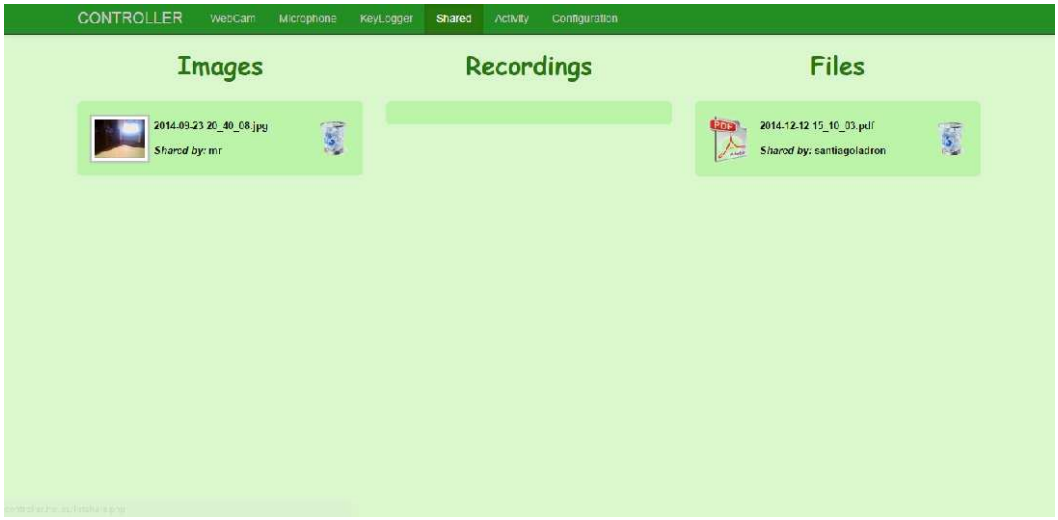


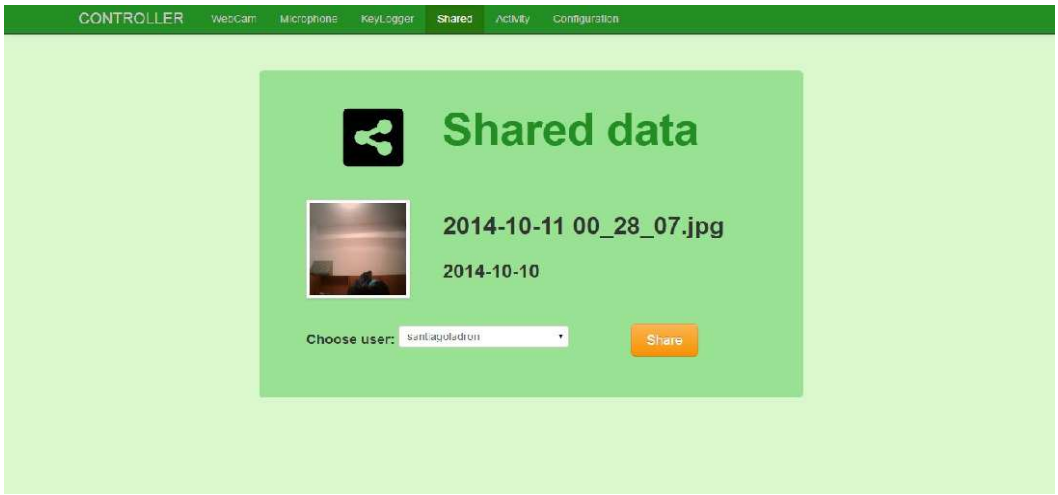
- Correcta interacción (update) con la base de datos a la hora de solicitud/cancelación de nueva petición y modificación de parámetros para cada uno de los distintos dispositivos. Prueba superada.



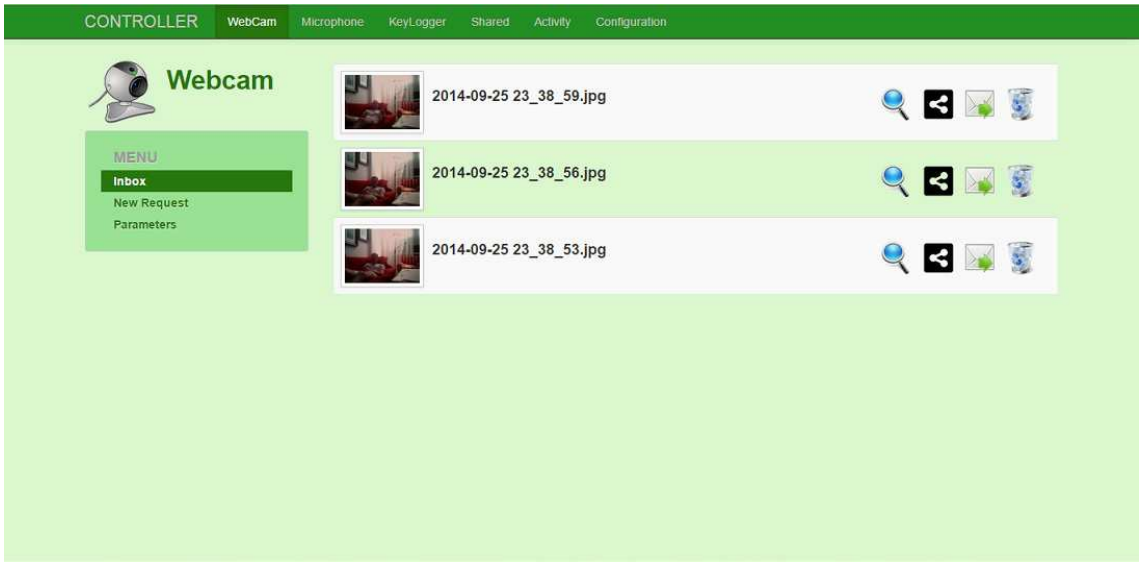
- Pruebas en la modificación de parámetros del dispositivo correspondiente. Aquellos campos que son de edición elegida por el usuario deben ser validados para comprobar que se ha insertado un valor de tipo correcto. Garantizar integridad de los datos. Prueba superada.

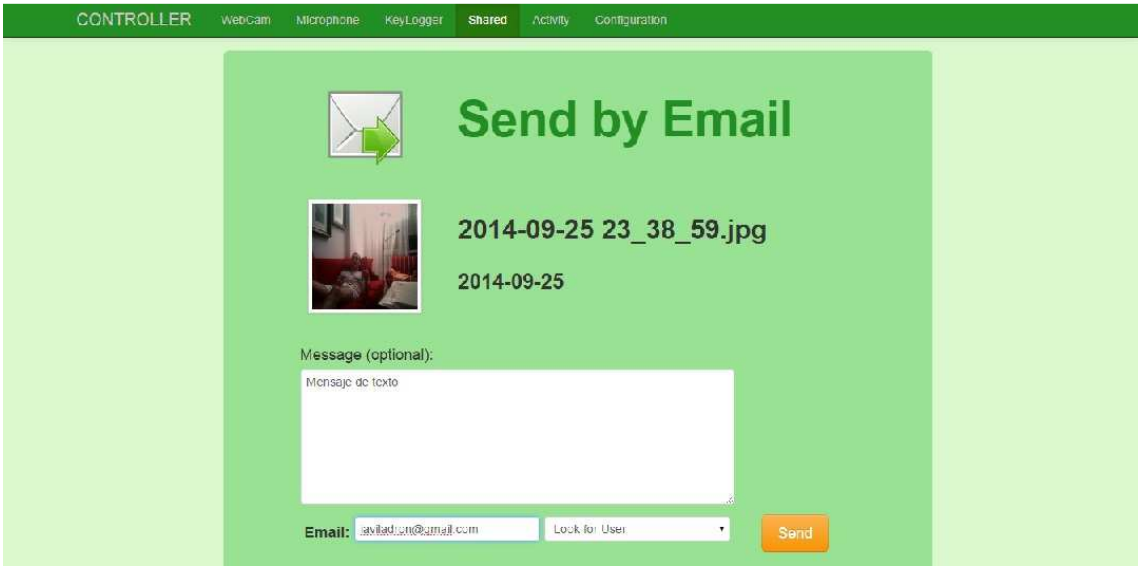
- Pruebas para la compartición de recursos entre usuarios del sistema. Correctas consultas, inserciones, borrados... con la base datos para llevar a cabo dicho servicio. En la zona de mostrado de recursos compartidos deben aparecer adecuadamente organizadas. Prueba superada.





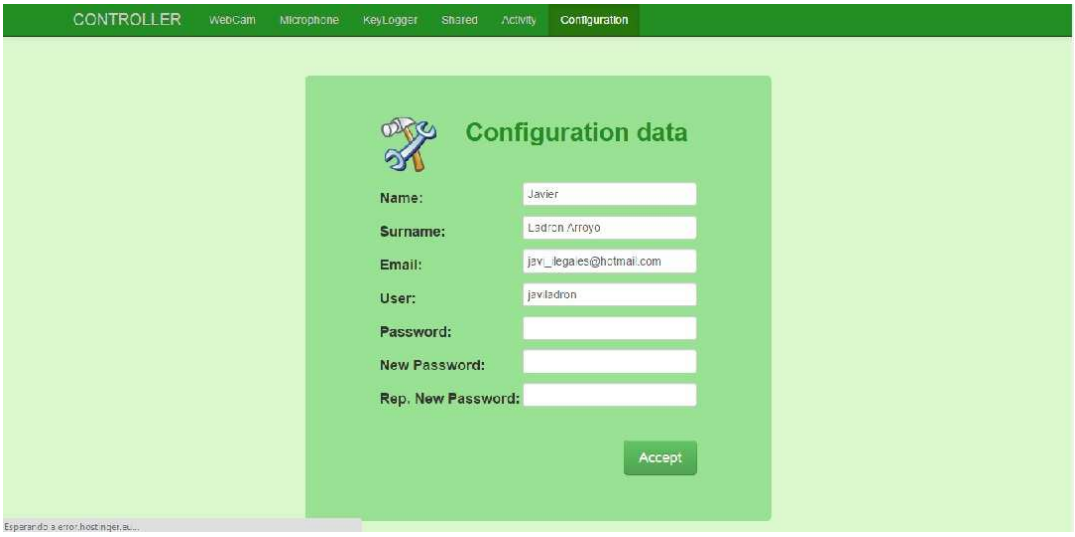
- Pruebas en el envío correcto de correos electrónicos con recursos del sistema. Consiste en pruebas con el servidor de correo que se quiere utilizar para el envío. En este caso, el servidor de correo de envío era de google bajo el protocolo smtp. Ningún problema. Prueba superada.





- Mostrado de la actividad generada por el sistema acorde a la especificación del proyecto. Consulta correcta. Prueba superada.

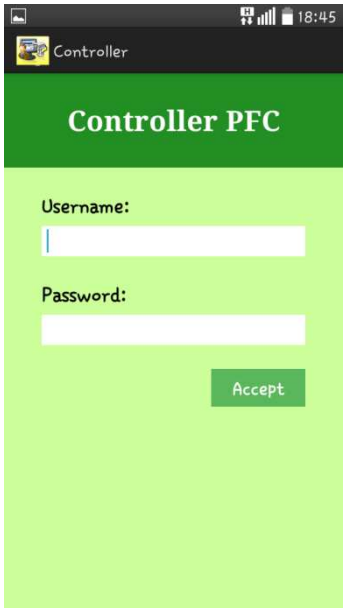
- Correcta modificación de los parámetros de usuario asegurándose siempre la validación de campos como formato email, coincidencias de contraseñas, campos texto o numéricos... Prueba superada.



Nota: toda interacción "persistente" (que conlleve una modificación en la base de datos) con la base de datos están tratadas con transacciones para garantizar la coherencia, cohesión e integridad de los datos en la base de datos.

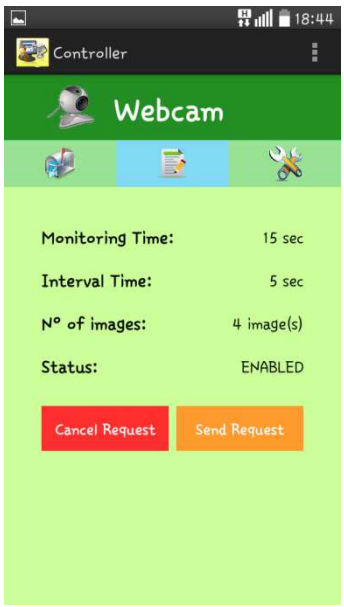
Pruebas unitarias en la aplicación Android.

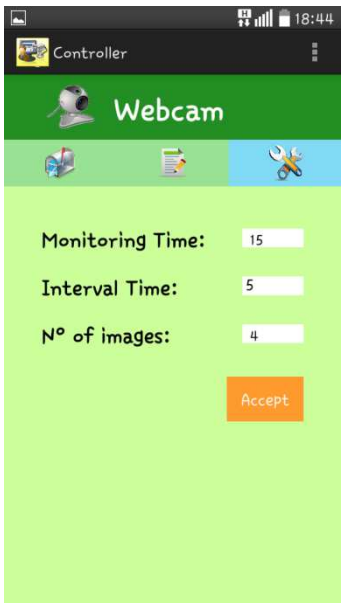
- Correcto uso de las SharedPreferences. Las utilizo para el sistema de autenticación automática para evitar al usuario el tener que loguearse cada vez que inicia la aplicación. Se aprovecha que es una memoria permanente (fichero xml). Prueba superada.



- Manejo de excepciones. Continua interacción con el servidor. Perdidas momentáneas de conexión, errores envíos de datos... pueden provocar fallos terminales en la aplicación. Se debe controlar todas esas posibles excepciones. Prueba superada.

- Control total de las conexiones con el servidor web para el funcionamiento correcto e la aplicación. Implementación correcta de clientes http respetando las condiciones del protocolo.





4.2 Pruebas de carga

En este apartado mostrare pruebas de carga que he realizado para las partes que conforman el sistema.

Pruebas de carga para el software controlador.

Antes que nada decir que conviene diferenciar el hecho de trabajar en local (misma máquina) o en una red de área local y hacerlo en Internet.
Las pruebas de carga que realicé en el software controlador fue en la subida de recursos al servidor. Cuando éste estaba en local o en la misma LAN aceptaba sin problemas el tope de recursos especificados en el sistema (10 imágenes en caso de la webcam, ficheros de audio y PDF de como máximo 60 segundos). La cosa es que a través de Internet, basado en las características del servidor, con más de 4 imágenes por ejemplo, a veces, ya se producía un algún error en alguna de ellas debido a la configuración del timeout del servidor.

Pruebas en el Remote Control Software.

Al igual que en anterior caso, las pruebas de carga vienen afectadas por la calidad de la conexión entre ambas partes del software. En este caso, la eficiencia de la red se mide por la cantidad de capturas de pantalla que se envían/reciben entre ambas partes correctamente frente a las que se desechan por exceso de congestión en la red.
De estas pruebas, surgió la decisión de incorporar la opción *calidad de servicio* (100%, 80%, 66% y 50%) para que en caso de que la red tenga problemas de flujo o congestión se disminuya la cantidad de bytes a enviar por la conexión o viceversa en caso de que las prestaciones de la red aumenten.

Pruebas en la página web.

En la página web las únicas pruebas que he podido han sido con la galería de imágenes mostrada en la zona de la webcam. Pueden ocurrir dos problemas. El primero de ellos con la cantidad de fotos en un día (forma de organizar recursos según la especificación) y el segundo con el tamaño, en bytes, de una foto.

Hay que tener en cuenta que los recursos, en este caso las imágenes están en el servidor, y se cargan online desde PHP. Ha pasado que una foto que pesa demasiado puede tardar un poco en salir.

En la aplicación Android no tiene problemas en cuanto a la carga. Todas las interaciones con el servidor reciben datos serializados en un string. Con lo cual no hay problemas de carga. Todos los problemas podrían venir por la calidad de conexión pero no por exceso de consumo de recursos.

5 Presupuesto

El siguiente paso es estimar el precio de lo que ha podido costar la realización completa del sistema.

Hay varios métodos, herramientas y mecanismos diferentes a la hora de valuar un proyecto. Pero al final me he decidido por evaluarlo según el número de horas empleadas hasta su final.

Hago una diferenciación entre el coste de la hora de análisis y diseño y entre el coste de la hora de desarrollo y pruebas.

El coste por hora de análisis y diseño del proyecto lo he valorado en 45€. En cambio el coste por hora de desarrollo lo he valorado en 38€.
Dicho esto y a partir de la estimación temporal, desglosando y evaluando cada una de las fases a partir de la semántica inicial espero obtener el coste total del proyecto.

Antes de empezar a evaluar estimo que el número al día son 4. Es una aproximación ya que hay días que se emplean más horas y otros días menos.

Lo haré por orden de realización, por lo cual empezaré antes con el análisis y diseño.
Esta parte está formada, por especificaciones, realización de diagramas, pseudocódigos...

Para el análisis se emplearon 4,5 semanas. En cambio para el apartado de diseño fueron 9 semanas .

13,5 semanas -> 94 días x 4 horas/día -> 376 x 45 €/hora -> 16920€

A continuación voy a valorar el periodo que me he pasado con la implementación a código fuente de los diseños hechos previamente.

El cálculo estará basado en 4 horas al día durante 17 semanas, que es el tiempo empleado estimado en codificar todo el sistema. Al ser un proyecto bastante amplio y con componentes es distintas plataformas me he visto obligado a emplear distintas tecnologías donde alguna de ellas las he tenido que aprender por mi cuenta.

A saber para la página web se ha empleado HTML, CSS, PHP y Javascript. El software controlador está implementado con .NET con el lenguaje C#. El software Remote Control esta desarrollado totalmente en Java y la aplicación en al ser para el sistema operativo Android está desarrollada con xml y Java. Por último la base de datos con SQL en un servidor MySQL.

Teniendo en cuenta que el precio por hora de desarrollo es de 38€ tenemos que:

17 semanas -> 119 días x 4 horas/día ->476 x 38 €/hora -> 18088€

Por último entraré a valorar el tiempo que he tenido que emplear en realizar las pruebas pertinentes para garantizar el funcionamiento correcto del sistema, corrigiendo posibles fallos que pudiese haber.

El precio por hora para las pruebas es el mismo que el del desarrollo, es decir, 38 euros.

El tiempo que se estima de pruebas para un proyecto es muy relativo, no se puede predecir con anterioridad. Pero una vez evaluado y probado el sistema por cada modulo por separado puedo decir que el tiempo empleado es de seis semanas. 42 días.

6 semanas -> 42 días x 4 horas/día -> 168 x 38 €/hora -> 6384€

En este presupuesto del proyecto no incluyo el coste por mantenimiento. Puede ser que por temas de servidor, calidad de de conexión de red pudiese haber ciertos problemas en el rendimiento del sistema. Pero en cuanto al código empleado en cada uno de los módulos es "cerrado". Se realizó todo lo que se describía en la descripción del problema.

Lo que sí debo añadir es el tiempo empleado en la realización de la documentación pertinente del sistema.
El precio hora para la realización es el mismo que el de la hora de desarrollo.

4 semanas -> 28 días x 4 horas/día -> 112 días x 38 €/hora -> 4256€

Por último vamos a realizar la suma de cada una de las partes para estimar el total del sistema. Dichas partes son análisis y diseño, desarrollo, pruebas y documentación.

Análisis y Diseño -> 16920 €
Implementación -> 18088 €
Pruebas -> 6384 €
Documentación -> 4256 €
+

Proyecto -> 45648 €

El proyecto está valorado en 45648 €

6 Conclusiones y líneas futuras

Se está llegando al final de la memoria y es hora de sacar algunas conclusiones.
Hablando a nivel personal, ha sido el proyecto más interesante, a la vez que largo que realizado hasta la fecha.

Merece la pena comentar que la idea de este proyecto se me ocurrió bastante antes de que empezara a abordar el problema de tener que realizar el Proyecto Fin de Carrera de Ingeniería Informática. Quise por mi cuenta hacer un sistema que inicialmente pudiese generar recursos de los principales dispositivos del ordenador y que fuese capaz de recibir peticiones remotamente. Según fui madurando la idea y viendo que podría ser un proyecto bastante interesante con algunas modificaciones y añadiendo mucho más contenido lo propuse para poder realizarlo y presentarlo como PFC.

Comentado cómo surgió la idea de realización me voy a disponer presentar las conclusiones pertinentes de cada una de las partes que conforman el sistema.

Lo primero es hablar del Software Controlador. Que por dar cierta relevancia a cada una de partes, es una de las dos partes clave/principales del proyecto.
La decisión en cuanto a tecnología a emplear, diseño para posterior desarrollo, interfaz gráfica, decisiones de comportamiento... fue una de las cosas que más tuve que pensar en el proyecto. Pero de todo lo comentado, una vez hecho el proyecto, me quedo con el hecho de haber tomado la decisión de desarrollarlo en una tecnología con la que no estaba familiarizado como era .NET con el lenguaje de programación C#.
Siempre es bueno aprender nuevas tecnologías y más si son punteras en su sector, de ahí me decisión.

La siguiente parte a comentar es la website que controla, organiza y presenta todo el sistema. Es la segunda de las dos partes clave/principales del proyecto. Obviamente hacía falta una herramienta que organizase, mandara producir o cancelar, permitiera manipular... todos los recursos que el Software Controlador es capaz de producir, además de llevar a cabo toda la gestión de protocolos de comunicación del Software Controlador con el servidor. Y como dicha herramienta debe estar completamente operativa en cualquier momento y desde cualquier terminal lo más fácil e inteligente fue por optar por una página web.

Del software controlador debo decir que fue un complemento que se diseñó y desarrolló cuando el resto del sistema estaba ya en las últimas fases de desarrollo.

Fue una herramienta que se me ocurrió dentro del trabajo de una asignatura de mi último año de carrera.
Quería un software de control remoto que tuviese una interfaz gráfica clara y precisa que permitiese al usuario el control remoto de otro terminal de manera amigable.

Y como se pretendía que este software, tanto cliente como servidor, fuese multiplataforma me decidí a hacerlo en Java.

Se puede ver como el proyecto utiliza varios lenguajes para intentar abarcar todas las tecnologías que actualmente se emplean en los diferentes sistemas operativos hoy en día.

Por último mostraré las conclusiones sacadas en la aplicación Android.

Fue lo último que se incorporó al sistema. Tiene dos motivos principales por los cuáles se ha desarrollado.

El primero de ellos es hacer una interfaz amigable. Se diseñó en su momento un diseño *responsive* y *adaptative* que fuera compatible con todos los navegadores de cualquier terminal. Pero siempre quedan mucho más estilizados y organizados los contenidos en una aplicación nativa.

Pero sobre todo, la desarrollé por temas de rendimiento y optimización. Clases java optimizadas para Android, procesos concurrentes para Android... hacen un cliente más rápido y funcional para este sistema operativo.

Con esto acabo las conclusiones de cada una de las partes. Puedo decir que ha sido el proyecto más entretenido que he realizado.
Además al ser un proyecto, que en un primer momento, fue todo una elección mía siempre se suele tener una motivación añadida ya que te gusta más que el hecho de realizar otro proyecto.

Ahora comentaré posibles líneas futuras que se pueden aplicar al proyecto.

Ahora mismo el software controlador es capaz de capturar fotos, grabar sonido y capturar teclas. Un cuarto servicio sería el de ofrecer la grabación de vídeo durante un determinado tiempo. Esta futura cuarta opción tendría como principal inconveniente el tamaño de este tipo de archivos. Hay que moverlo a un servidor en Internet minimizando retardos y pérdidas. Habría que asegurarse de elegir una librería que permitiera un formato de compresión adecuado de vídeo.

Otro tema de mejora del Software Controlador es el parser integrado sobre terrorismo (en español) que posee.

El parser que está ahora mismo se puede mejorar monitorizando más verbos sospechosos (todos los tiempos verbales de cada uno), aumentando el número de palabras clave (sustantivos) para poder abarcar más sobre el tema.
Con estas modificaciones en la gramática, analizaría más casos. Pero quiero dejar claro, que este tema que menciono lo que se haría es añadir más contenido sobre una base (en la gramática) ya realizada. La estructura ya está hecha en esta versión.
Lo que siempre se puede hacer es proponer otras gramáticas para capturar patrones sobre otros temas relevantes como economía, situación laboral, casos de pornografía...
O simplemente el hecho de una gramática que permitiese el reconocimiento de patrones sobre terrorismo u otros temas pero en diferentes idiomas.

En el software Remote Control hay varias cosas que pueden evolucionar en un futuro
La primera de ellas es el sistema de autenticación del server mediante identificación y reconocimiento facial.
Ahora mismo existe una versión beta en la que el sistema es capaz de permitir el acceso a una cara (imagen estática en una foto) a partir de un entrenamiento sobre esa cara. A su vez rechaza el acceso al sistema de otra cara distinta (otra imagen estática en una foto).
A lo que se pretende llegar es a un sistema en el que la primera vez que el usuario quiera acceder al software desarrolle un sistema de entrenamiento automático en el que capture todas las imágenes pertinentes. Teniendo en cuenta detalles como distancia, luminosidad, perfiles, barba... y después de ejecutar el algoritmo pertinente (PCA) se consiga una serie de

eigenvalues y eigenvectores con el mínimo error posible para garantizar los mínimos problemas a la hora de autentificar el rostro.
Dicho esto, el sistema también debería, en el momento de la autentificación tomar una foto en ese instante y comparar y evaluar, según el entrenamiento realizado si esa persona tiene derecho o no al acceso al servidor.

Otro tema relevante en el que se puede trabajar en el Software Controlador es en la representación del puntero remoto en el cliente del software.
Ahora mismo en el software, si los dos monitores de los terminales donde se ejecutan cliente y servidor, tienen distinta resolución hay ciertas (mínimas) variaciones en la representación de la posición del puntero remoto.

La última cosa que me gustaría mejorar del Remote Control es el protocolo de conexión entre ambas partes del sistema.
Ahora mismo el protocolo que utilizo es propio, bajo conexiones TCP, donde se utilizan, puertos 8000 y 8001 en el servidor para permitir conexiones con el cliente. Esto permite que, en una misma LAN cerrada no exista ninguna restricción ni problema de conexión. Pero a través de Internet, cuando ambas partes están en redes distintas (privadas claramente) con varios routers de por medio hay que configurar varias cosas en el router del server para poder saltarse NATs, Firewalls...
Debido a esto, se deben encontrar e implementar soluciones que eviten todos estos problemas. Técnicas basadas en VPN, UDP Hole Punching (técnica del software remoto profesional TeamViewer), servidor proxy público... podrían resolver el problema planteado.

De la aplicación Android se puede completar el hecho de poder visualizar los recursos directamente desde el servidor pero representados en la app de forma amigable, clara y concisa. Ahora mismo, en la app, todo recurso generado por el sistema se puede descargar del servidor y luego se puede visualizar, reproducir... con otras herramientas del sistema operativo.

De la página web habría que considerar el hecho de adaptar secciones y opciones para abarcar futuras ampliaciones de otras partes del sistema pero que se verían reflejadas en la web. Por ejemplo, si en el Software Controlador se hiciera la opción de captura de vídeo habría que diseñar y desarrollar una zona en la web que permitiese el control y gestión de este nuevo tipo de recurso.

También se podría completar la web con animaciones, optimizadores de carga mediante lenguajes de cliente y librerías (JavaScript, AngularJS, JQuery, AJAX...).

7 Bibliografía

[1] Documentación sobre el lenguaje de programación orientado a objectos. http://www3.uji.es/~belfern/pdidoc/IX26/Documentos/introJava.pdf

[2] Documentación sobre compiladores y procesadores del lenguaje. http://dinosaur.compilertools.net/

[3] Documentación a modo de introducción sobre las características básicas de .NET https://msdn.microsoft.com/es-es/library/4w3ex9c2%28v=vs.100%29.aspx

[4] Documentación oficial sobre Objetive C, el lenguaje de programación utilizado en el sistema operativo iOS.
https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html

[5] Documentación sobre el framework .NET y sobre el lenguaje de programación C# http://msdn.microsoft.com

[6] Documentación sobre el lenguaje C# para el framework Microsoft .NET https://msdn.microsoft.com/es-es/library/z1zx9t92.aspx

[7] Documentación sobre compiladores. Información obtenida de la materia Procesadores del Lenguaje, 4º Ingeniería Informática, UPNA.

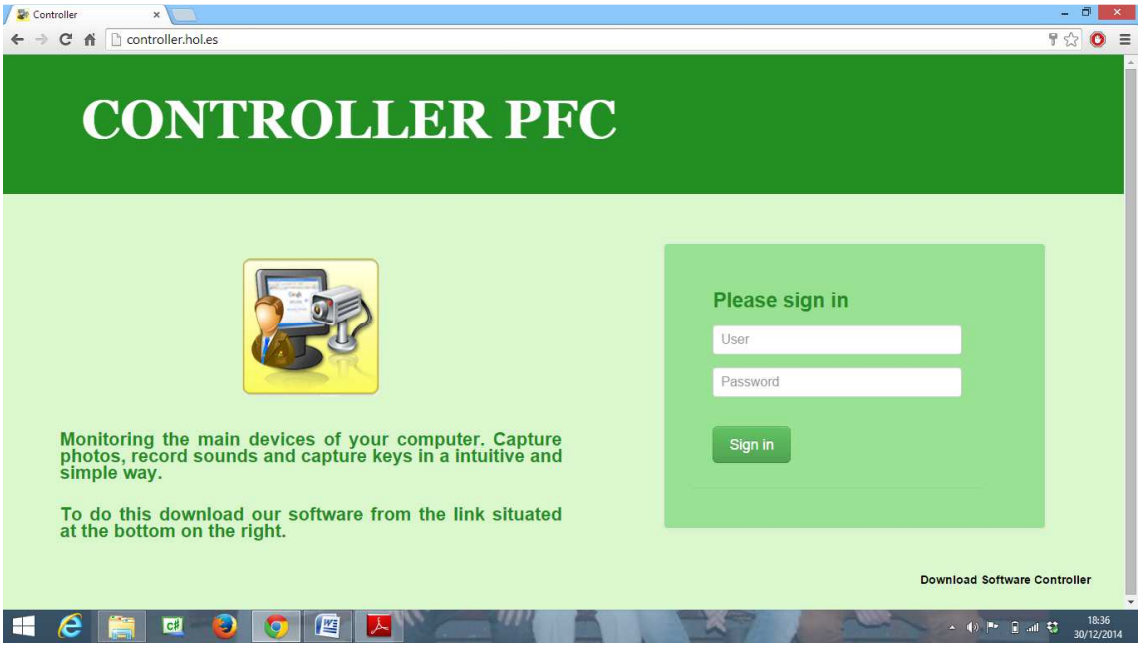
[8] Documentación sobre la herramienta GOLD Parser Builder para la construcción de parsers LALR. Información obtenida de http://goldparser.org

[9] Documentación sobre la librería de visión por computador OpenCV. Información obtenida de http://docs.opencv.org

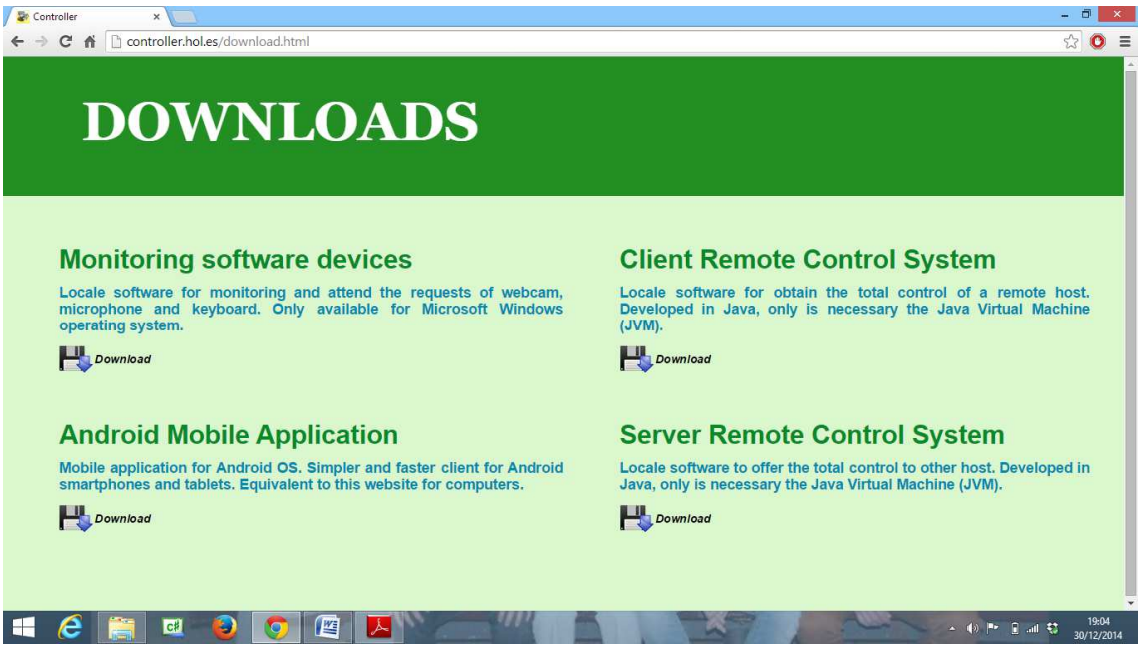
8 Anexo I. Manual de usuario

En este apartado voy a realizar el manual de usuario pertinente para todas las partes del sistema. Apartado de realización obligada para aclarar todas las posibles dudas que le pudiesen surgir al usuario de cada uno de los módulos o partes.

En primer lugar realizaré el manual de usuario para la página web.



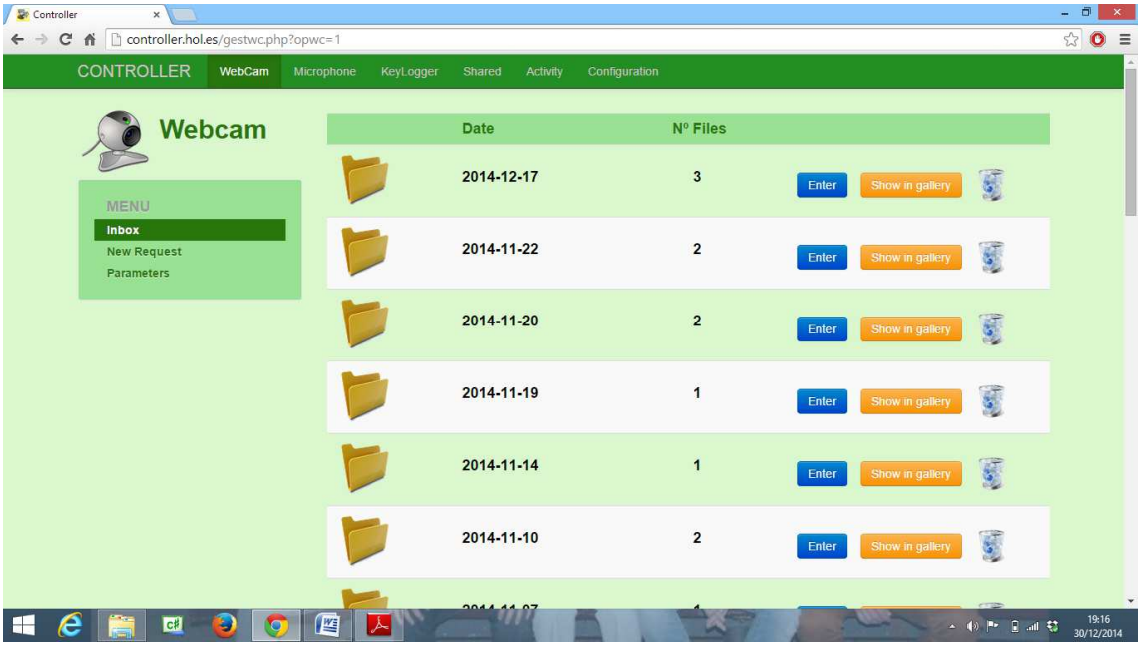
Se ve como en el index hay una zona para la autenticación del usuario mediante nombre de usuario y contraseña. A su vez, abajo a la derecha se muestra un enlace para obtener la siguiente pantalla con todos los softwares descargables que forman el sistema.



Una vez autentificado el usuario correctamente la web muestra una interfaz con un menú principal formado por: Webcam, Microphone, Keylogger, Shared, Activity, Configuration.

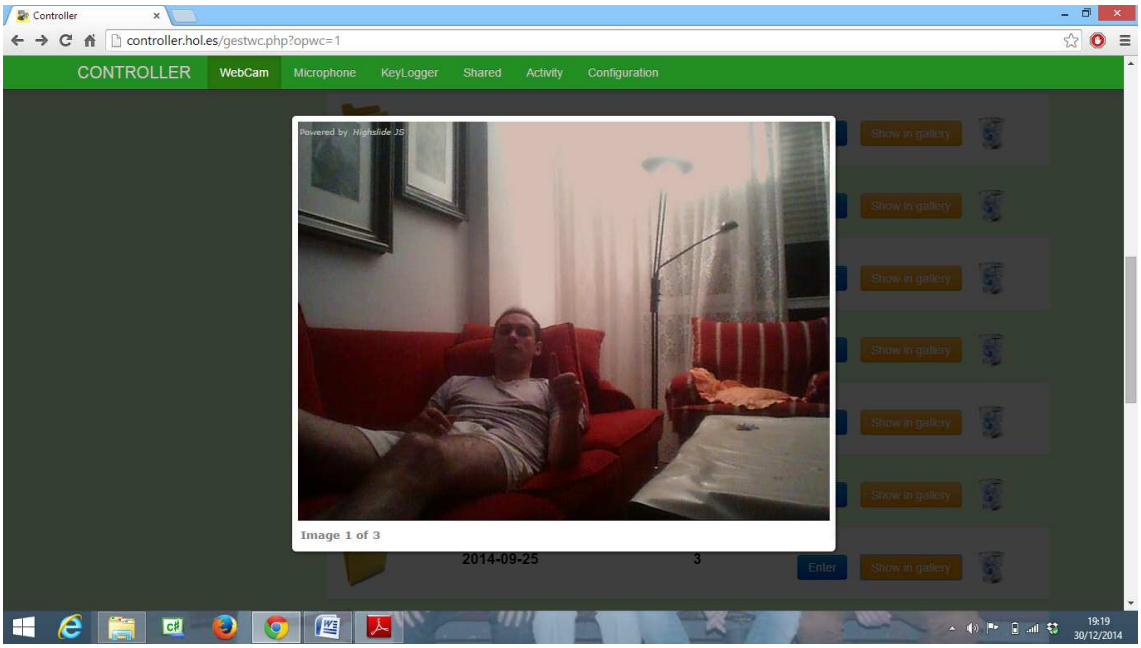
Mostraré a continuación las pantallas de las bandejas de entrada de cada uno de los 3 dispositivos donde se podrá apreciar cómo se organizan cada uno de los recursos y que opciones se pueden realizar con ellos.

Veamos cómo se muestran las imágenes de la webcam.

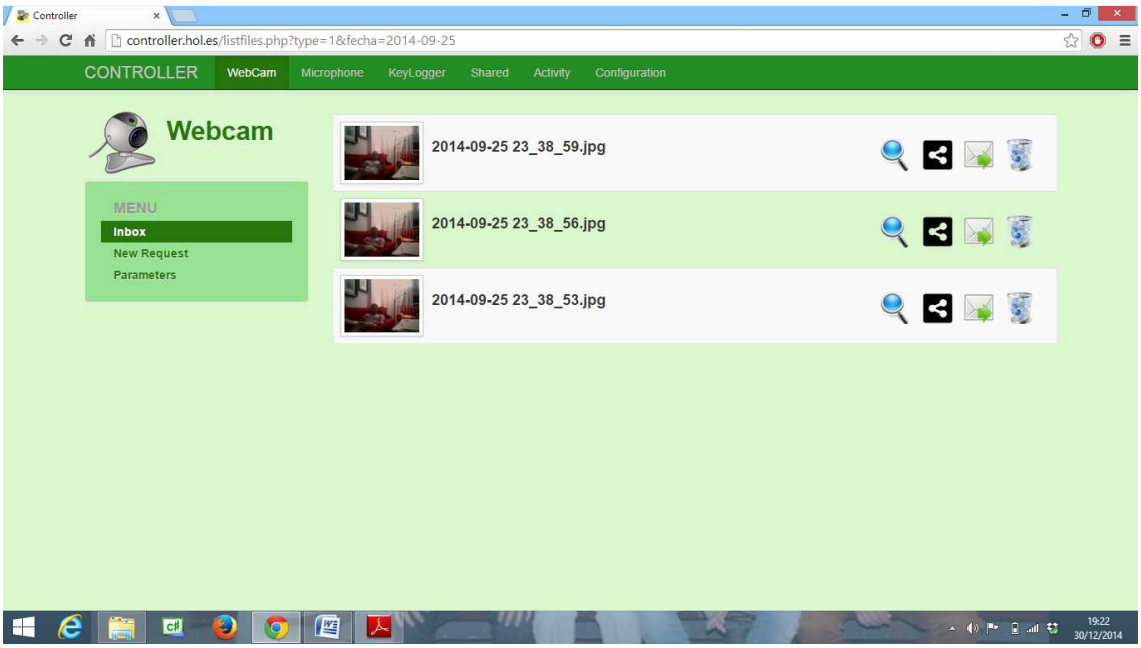


Se pueden ver cómo están organizadas por días. Pulsando sobre la papelera eliminas todas las imágenes de ese día.

Pulsando sobre el botón naranja *Show in gallery* se muestra una galería con opciones de play, next, previous... de las imágenes de ese día.

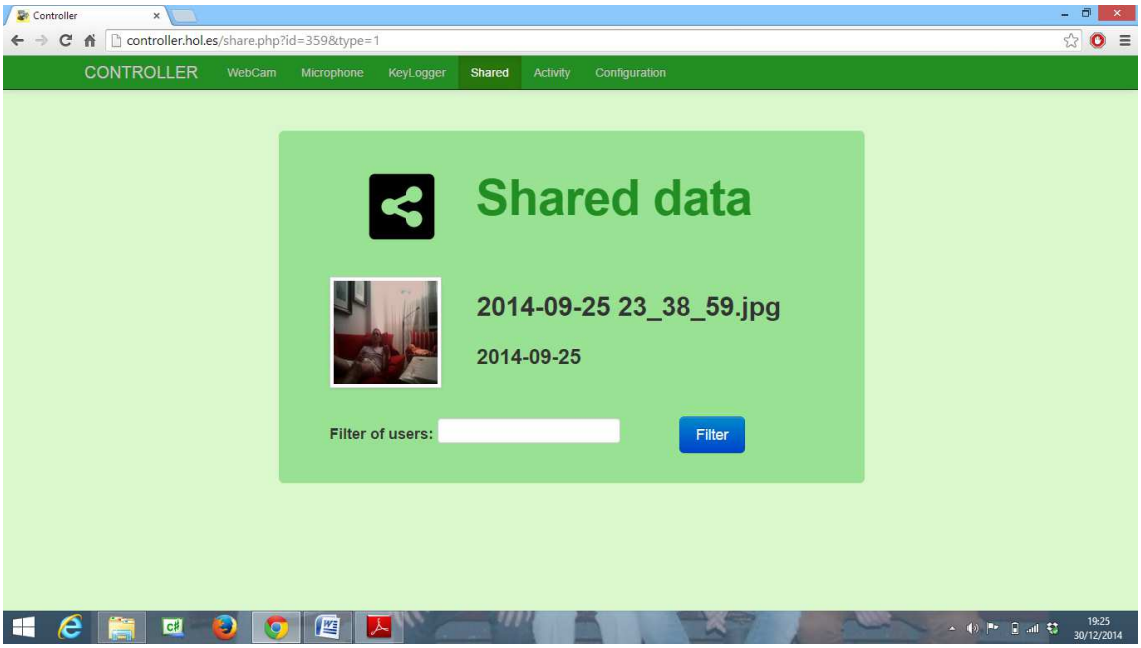


Pulsando sobre el botón azul *Enter* vamos a una pantalla donde se muestra un listado de cada una de las imágenes con diferentes opciones.

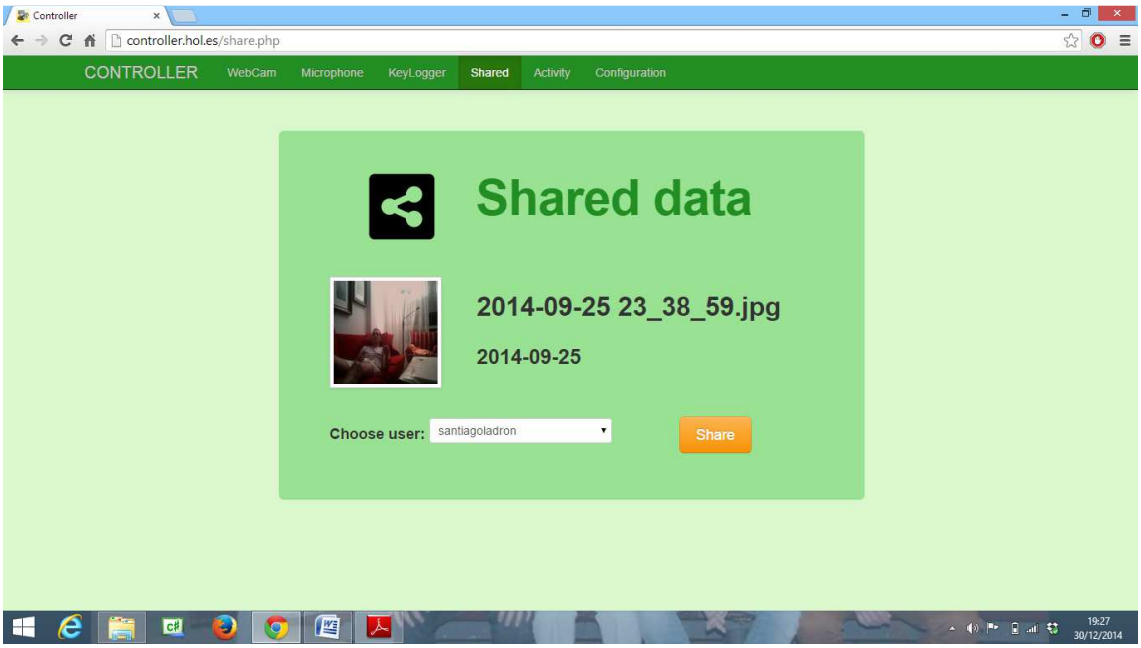


Vemos una lupa donde se puede visualizar individualmente dicha imagen. También hay una papelera donde se puede eliminar cada imagen individualmente.

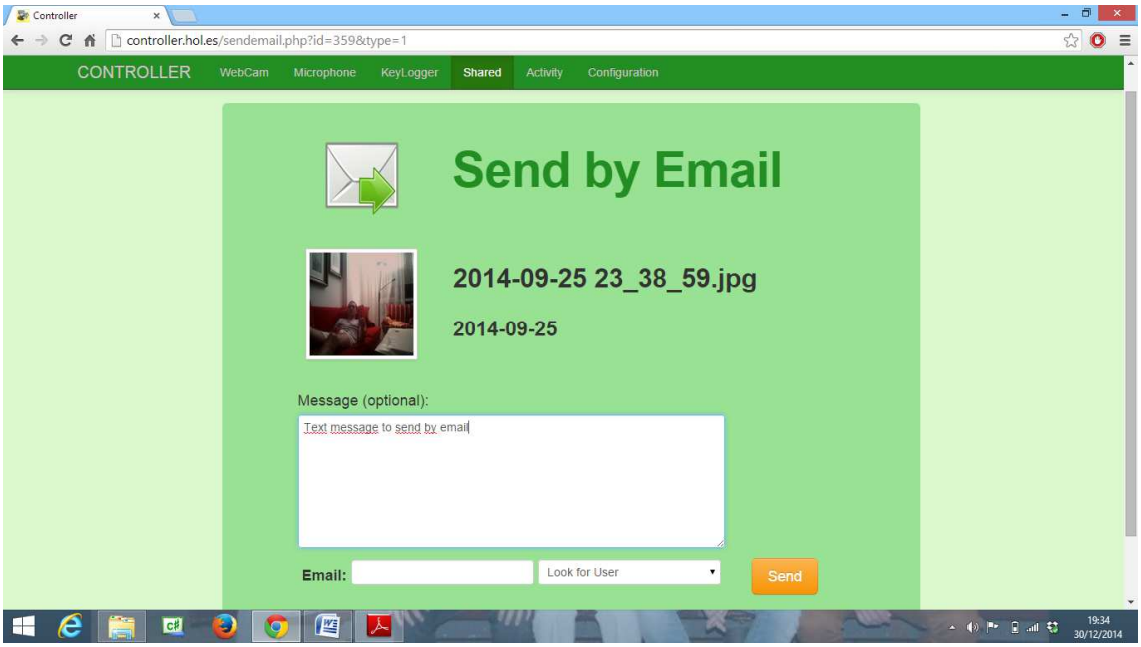
Se puede ver también como existe un botón de *compartir* donde se puede compartir con otro usuario del sistema la imagen para que aparezca en la zona de Shared.



En la siguiente pantalla se ve como introduciendo un patrón de texto en el textbox el sistema busca posibles usuarios que coincidan para que el usuario pueda elegir el deseado y compartir la imagen.

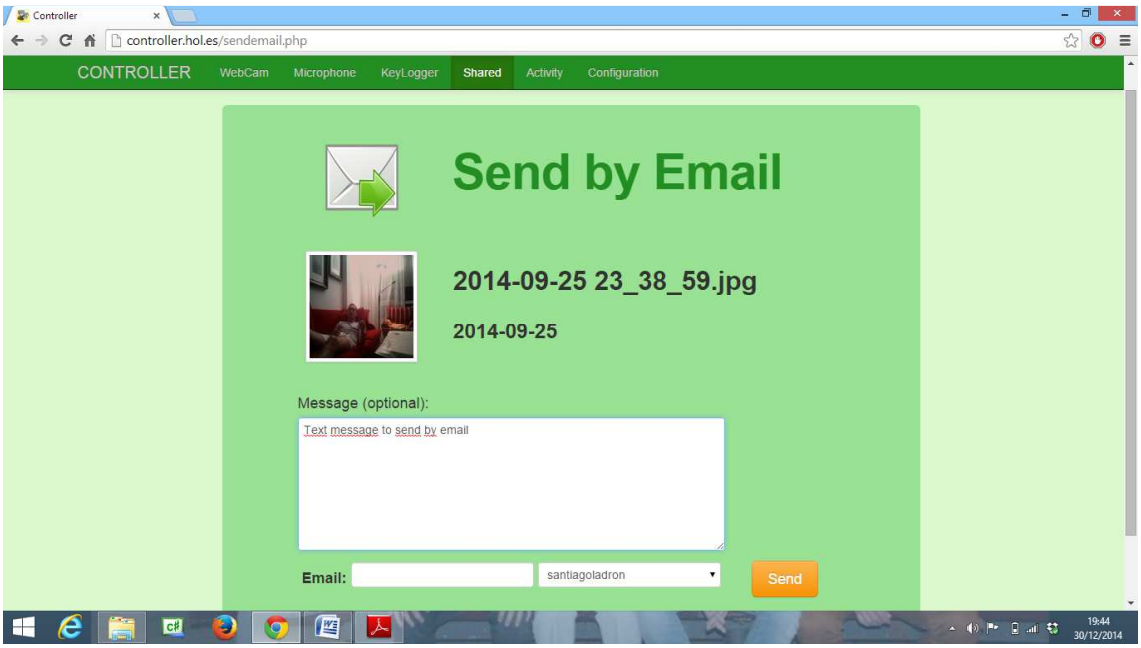


Con la función de correo electrónico ocurre bastante parecedido.



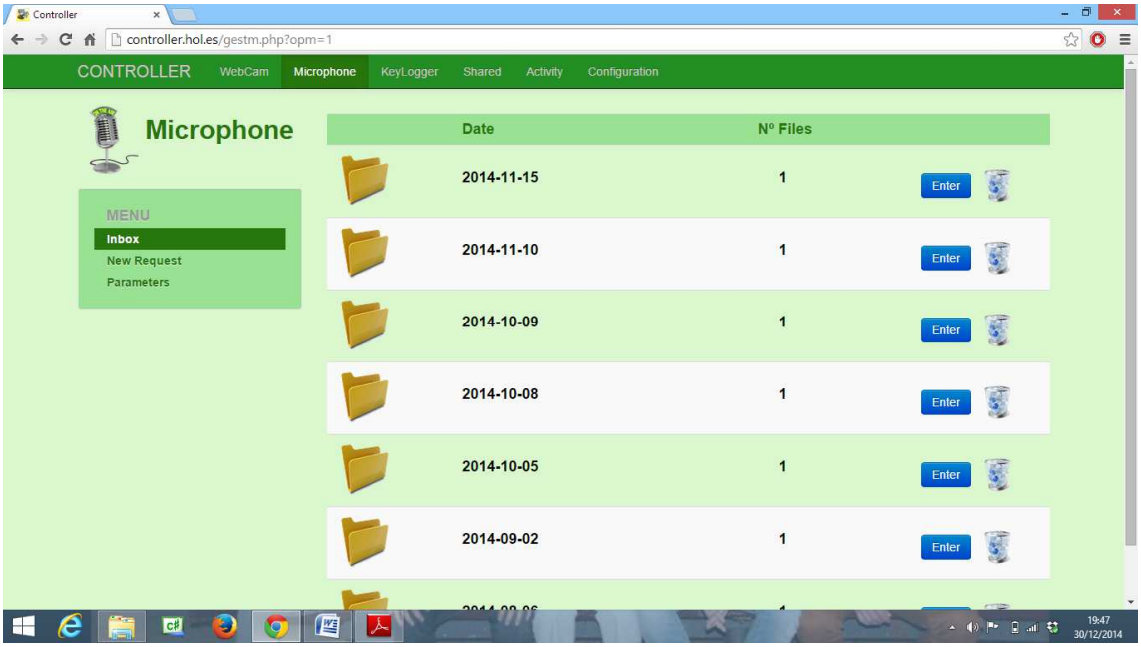
Se puede ver como el usuario puede elegir entre, directamente, enviar a una dirección de email válida o, en cambio, introducir un patrón de texto, con el cual el sistema mostrará aquellos emails registrados validos que se adecuen al patrón introducido.

En la siguiente pantalla se muestra el resultado previo al envío del email si el usuario ha optado por la segunda vía.



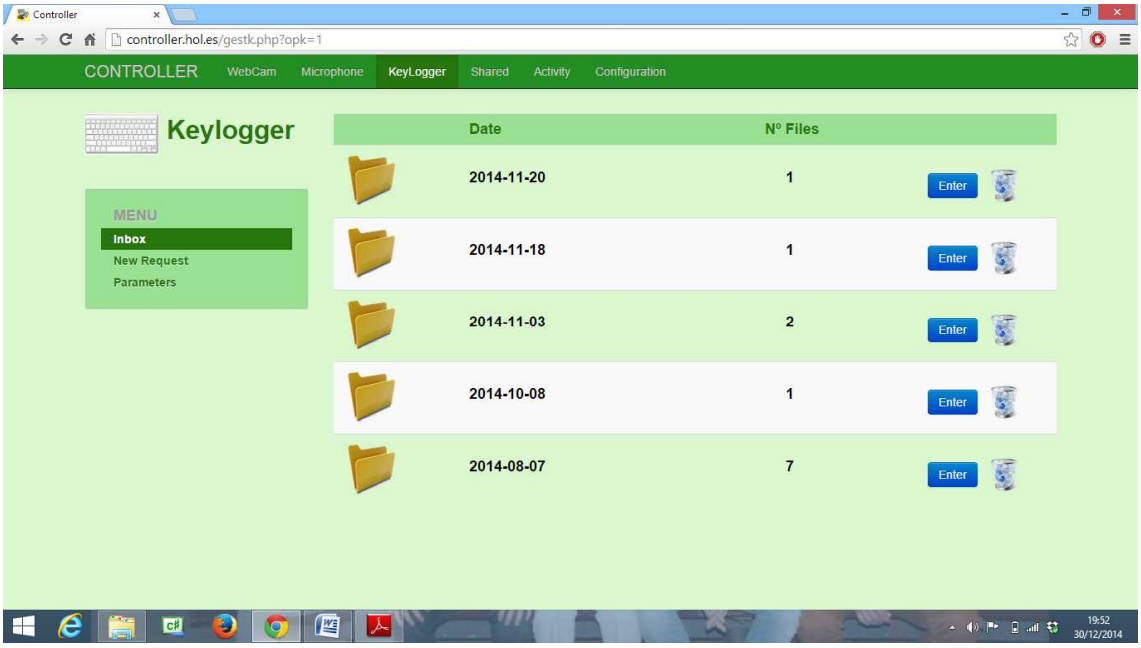
A continuación mostraré las interfaces de bandeja de entrada de Microphone y Keylogger . No mostraré el detalle de opciones (compartir, enviar por email...) de los recursos grabaciones y ficheros porque es exactamente igual que para las imágenes de la webcam. La única diferencia es que para cada grabación en lugar de visualizar se puede escuchar online y para cada fichero se puede descargar en lugar de visualizar.

Interfaz para Microphone:



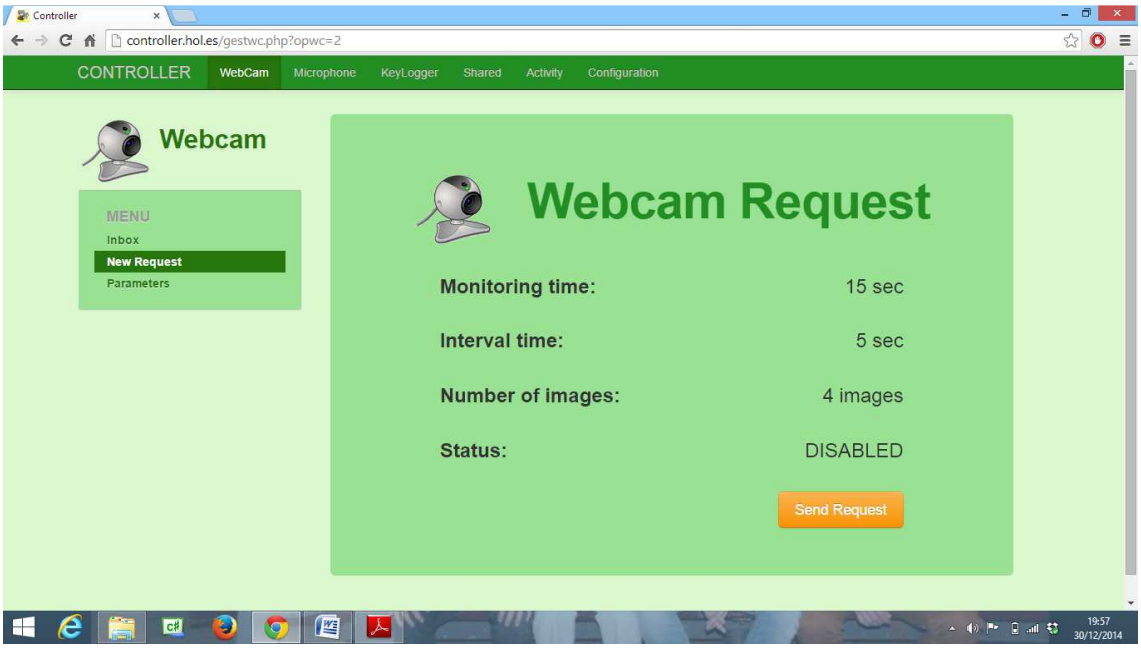
Se ve como hay dos opciones para cada uno de los directorios diarios de grabaciones. La de eliminar todas las grabaciones de ese día y un botón *Enter* donde, al igual

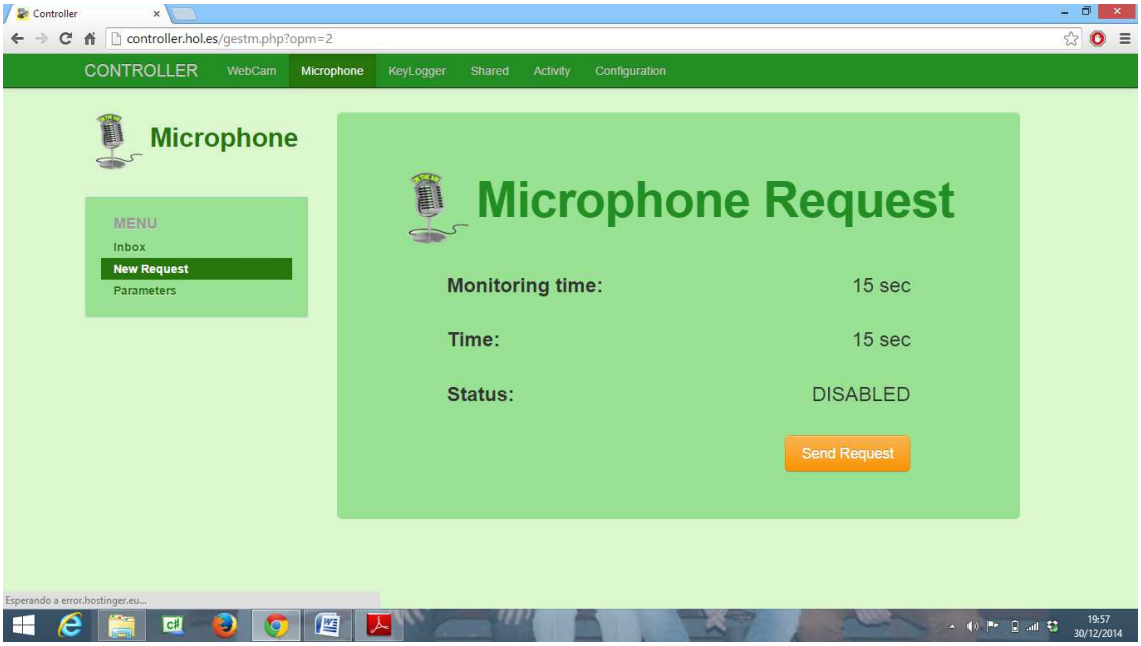
que en el caso de la Webcam, muestra una vista listado con diferentes opciones para cada una de las grabaciones de ese día.
Interfaz para Keylogger

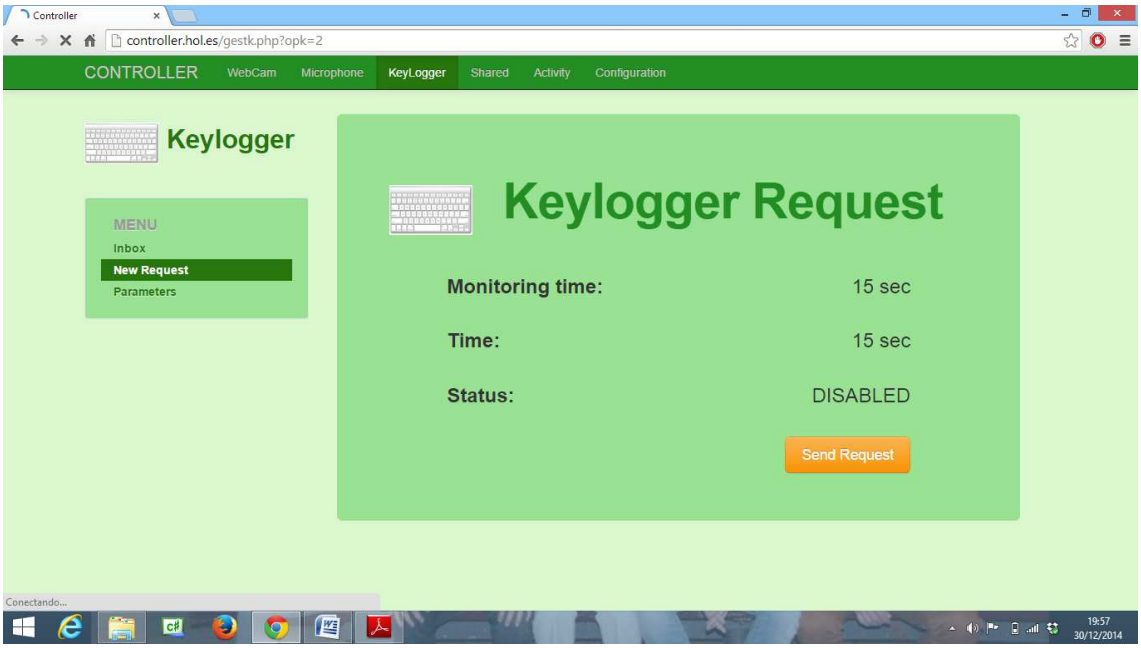


Se ve como hay dos opciones para cada uno de los directorios diarios de ficheros. La de eliminar todas los ficheros de ese día y un botón *Enter* donde, al igual que en el caso de la Webcam y Microphone, muestra una vista listado con diferentes opciones para cada uno de los ficheros de ese día.

A continuación voy a ilustrar los interfaces de cada uno de los dispositivos donde se puede enviar o cancelar una nueva petición al Sistema Controlador.



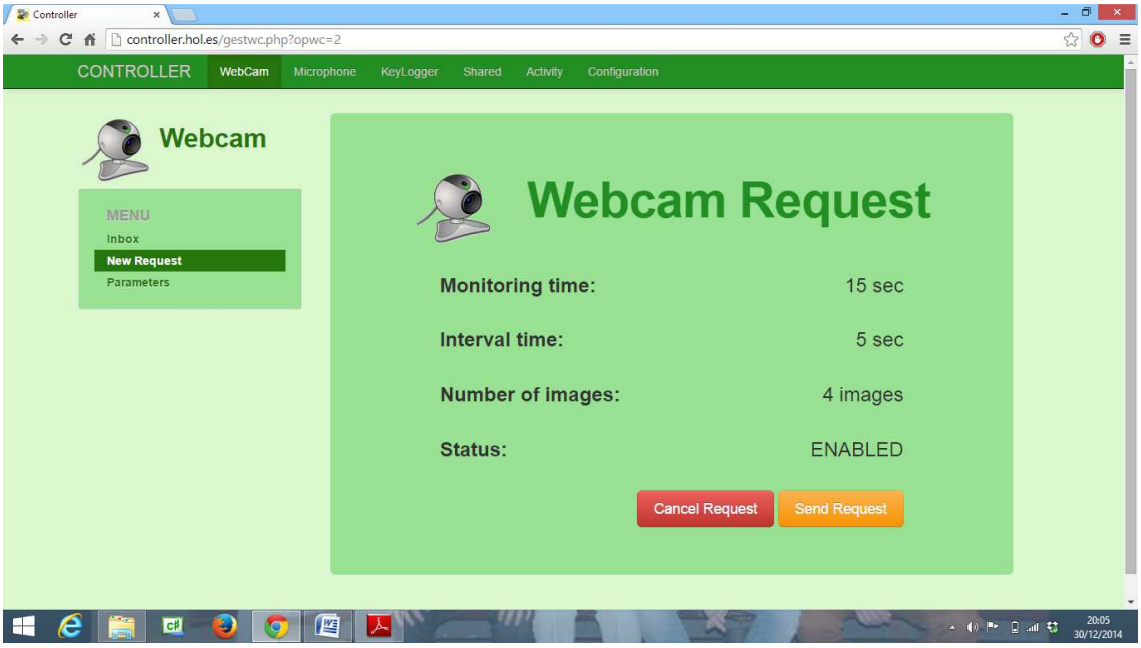




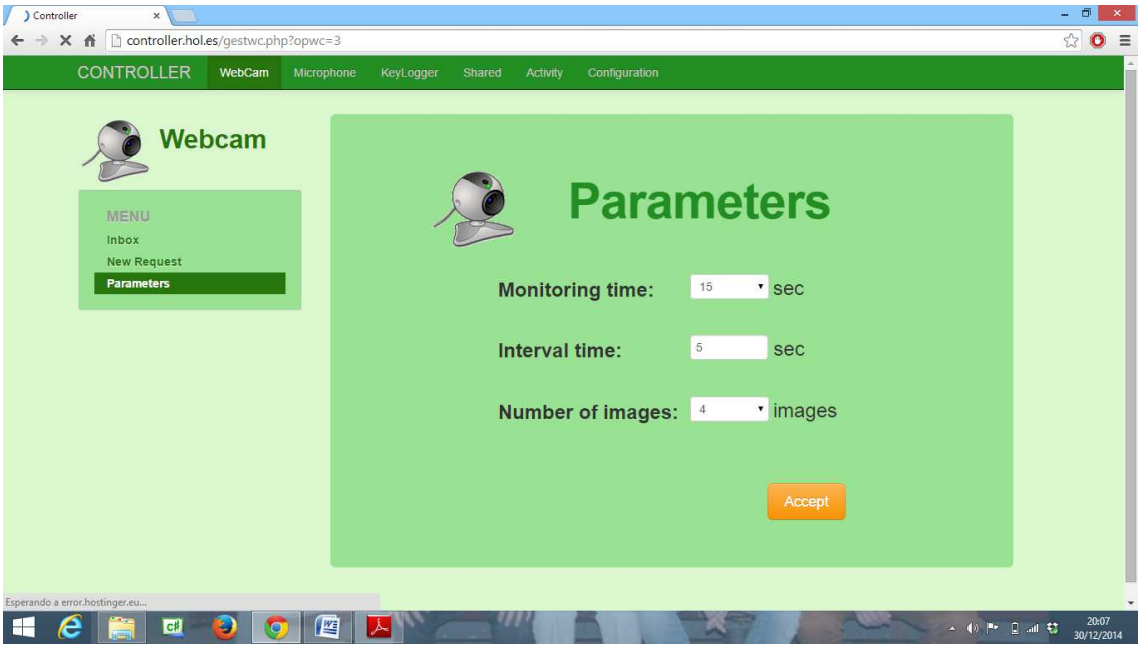
Se ve, en cada uno de los tres últimos interfaces , los parámetros con sus determinados valores de los diferentes dispositivos.

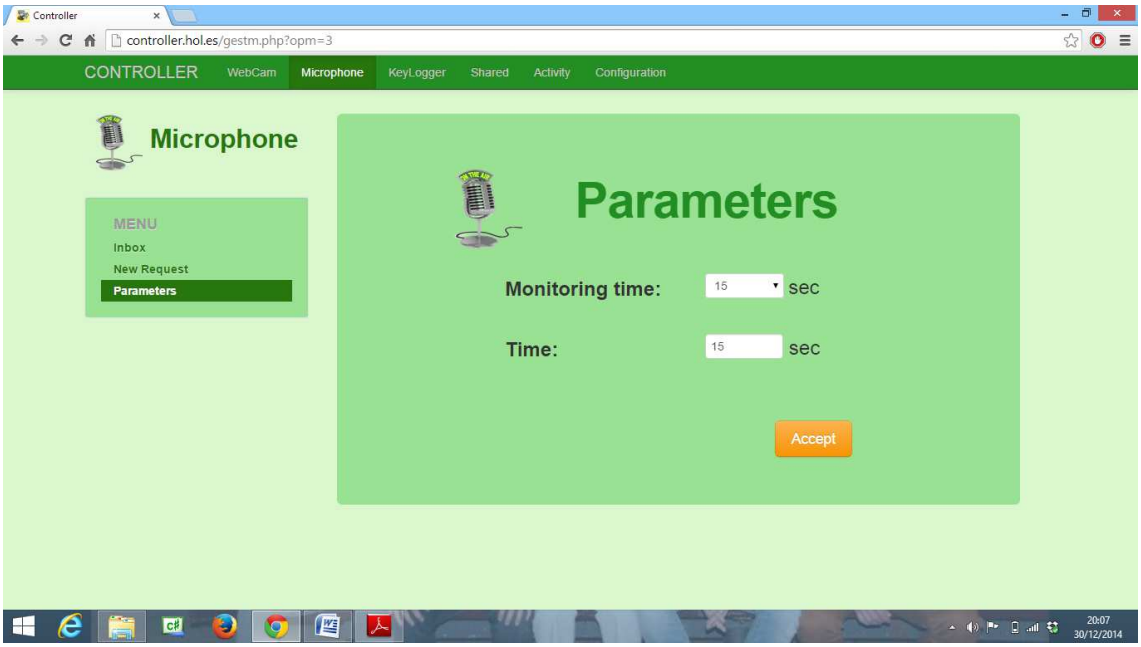
Se muestra también un botón naranja *Send Request* que es que envía la nueva petición al sistema.

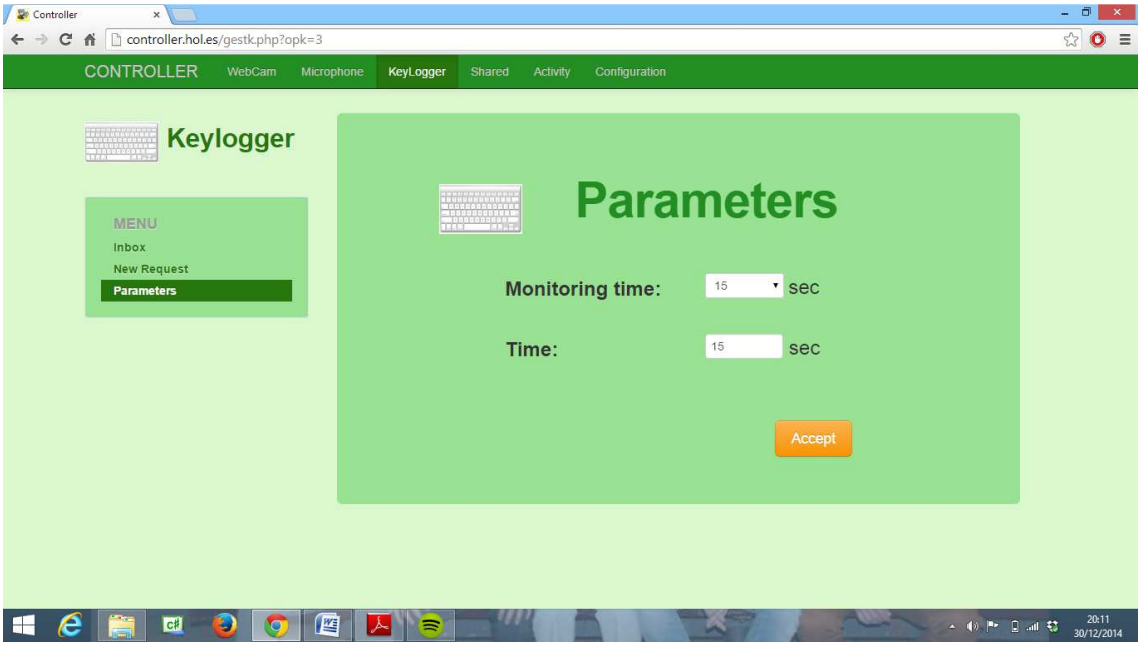
En el siguiente pantallazo se muestra un ejemplo de que una vez enviada a petición al sistema se da como opción inmediata la posibilidad de cancelar la petición previamente solicitada. Funciona igual para los tres dispositivos. Botón *Cancel Request*.



A continuación mostraré los interfaces, para cada uno de los dispositivos, de la modificación de parámetros.



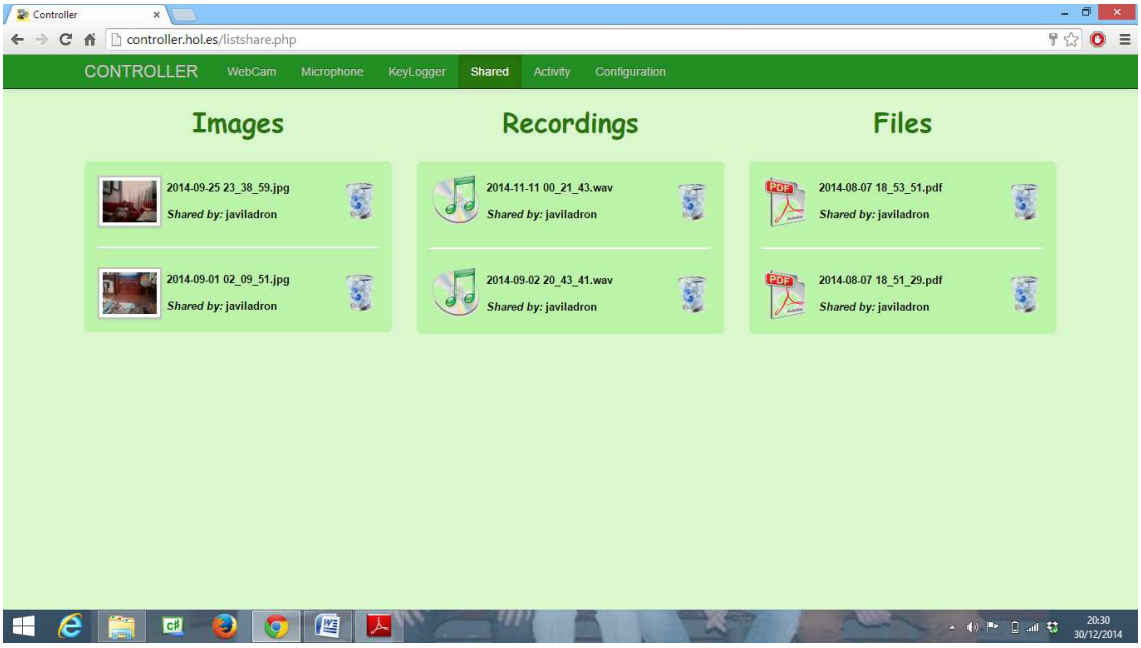




Se puede apreciar como con el botón *Accept* modifica cada uno de los conjuntos de parámetros de los dispositivos.

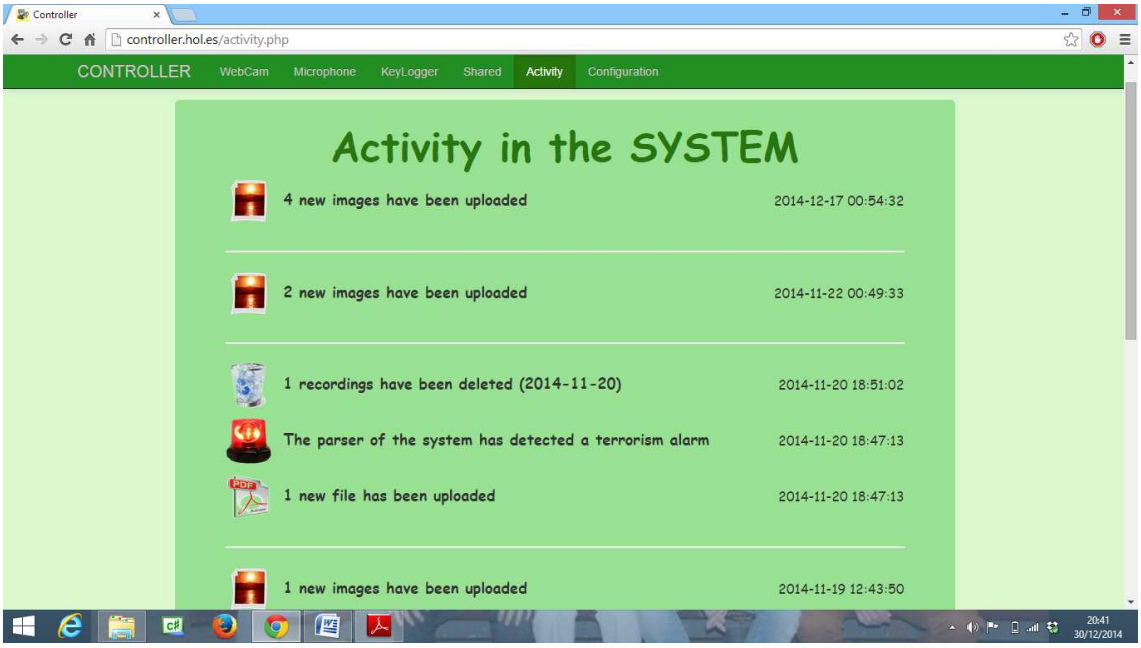
Esta modificación de parámetros es exactamente la misma que existe en el Software Controlador.

A continuación, en el manual de usuario, mostraré el interfaz donde se muestra todos los recursos compartidos por otros usuarios del sistema con el usuario.
Se divide la pantalla en 3 zonas, cada una para un distinto tipo de dispositivo.

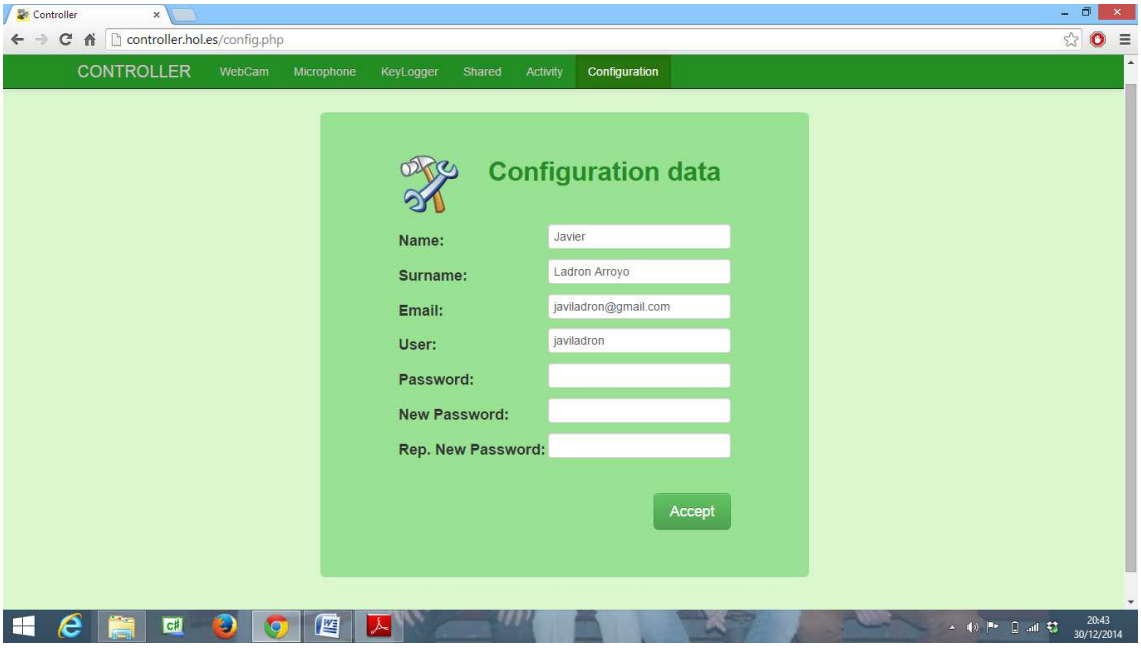


En este caso se ven dos recursos compartidos de cada recurso. De cada uno de ellos se muestra el nombre y el usuario del sistema que ha compartido el recurso.
Para imágenes se puede visualizar, para grabaciones reproducir y para ficheros descargar.
Además se puede eliminar individualmente cada uno de ellos con la papelera.

El siguiente interfaz a mostrar será el de la actividad generada por el usuario en el sistema donde este puede ver reflejado todo lo que ha realizado.
Tiene una opción de borrar historial si desea borrar todo el histórico.



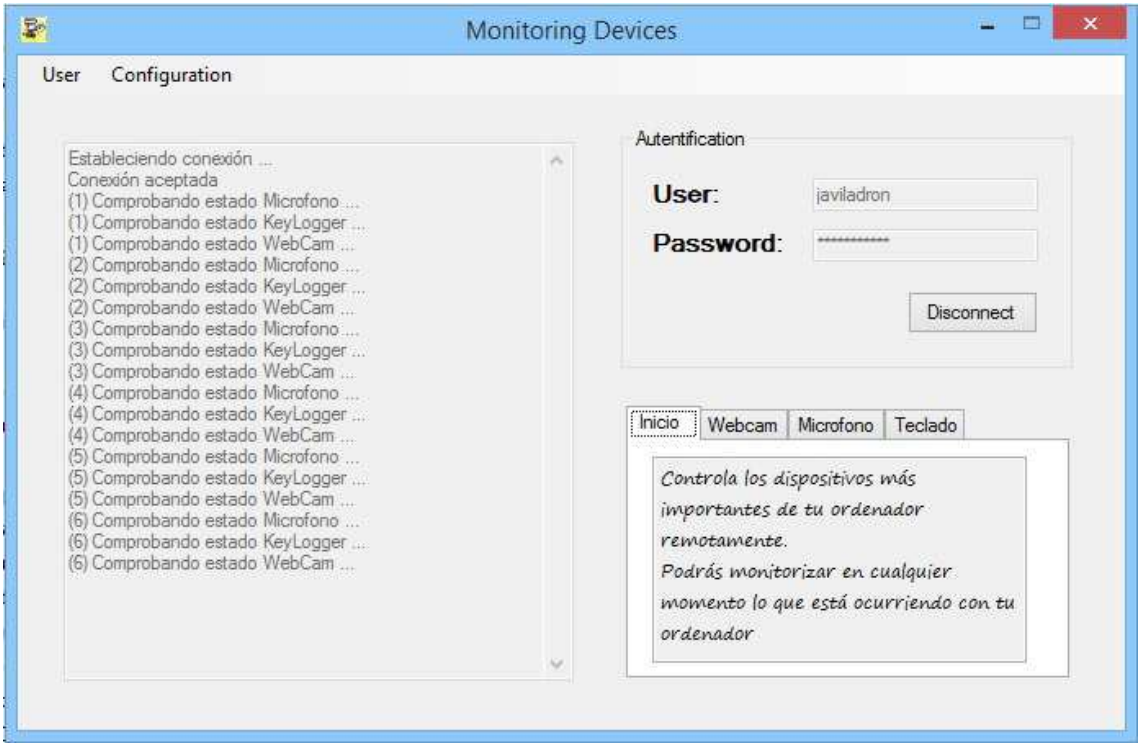
Por último, en la web, mostraré el panel de configuración donde el usuario puede modificar sus datos personales.



Todos los campos están a pruebas de posibles fallos de usuario. Además de adoptar las típicas normas de seguridad como no mostrado de contraseñas, repeticiones...

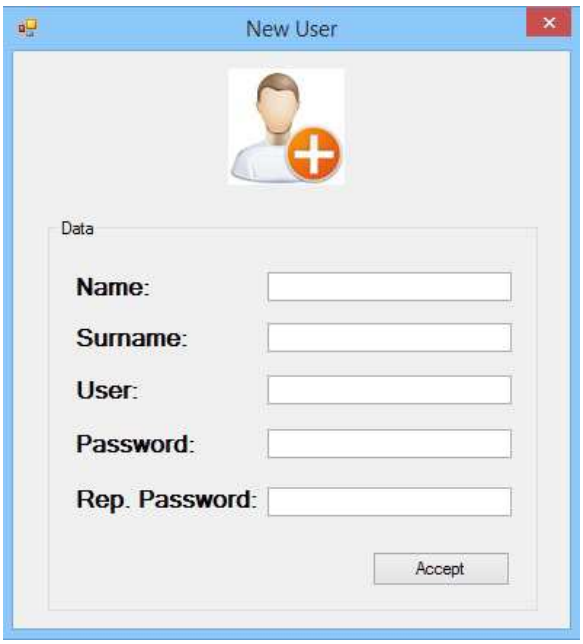
La siguiente parte del manual del usuario será para explicar el interfaz gráfico del Software Controlador.

En la interfaz principal se puede ver una zona de autenticación donde, mediante usuario y contraseña correctos, se activa el software.

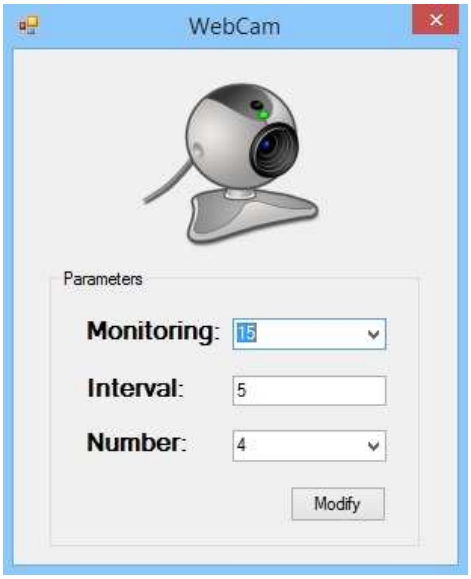


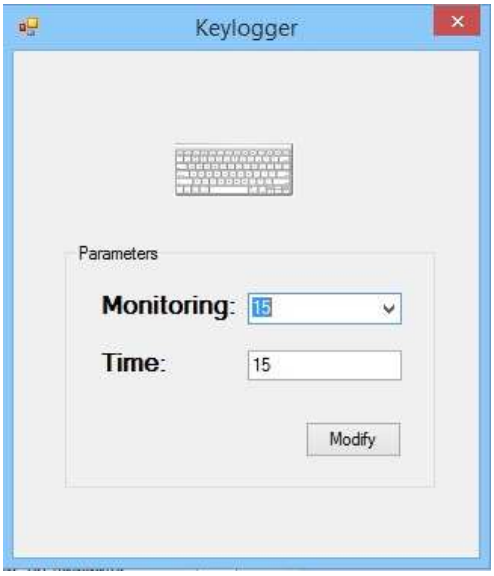
En la pantalla de la izquierda se ve reflejado una serie de mensajes en un display. Estos mensajes detallan la monitorización del estado de los dispositivos en el servidor web.

La siguiente pantalla es la nuevo registro de usuario en el Software Controlador. Debe ser utilizada en la primera ejecución el software o después de borrar usuario.



A continuación mostraré los 3 paneles de configuración de parámetros de cada uno de los diferentes dispositivos.
Estos paneles solo son accesibles una vez que el usuario se ha autenticado correctamente.





Nota: Estos paneles permiten modificar los parámetros de monitorización de los dispositivos de igual manera que se podía hacer en la web.

El siguiente apartado a mostrar será el funcionamiento del Remote Control Software.

En primer lugar mostraré el servidor.

Dejando de lado la beta de autenticación por detección y reconocimiento facial, el funcionamiento del servidor es bastante sencillo. Basta con pulsar el botón de abajo a la derecha para empezar o parar el servicio.

Se puede ver la ip de la LAN que tiene, el puerto donde "escucha" el servicio, el estado del servidor "Activate" o "Desactivate" y la cantidad de bytes que se transfieren al cliente.

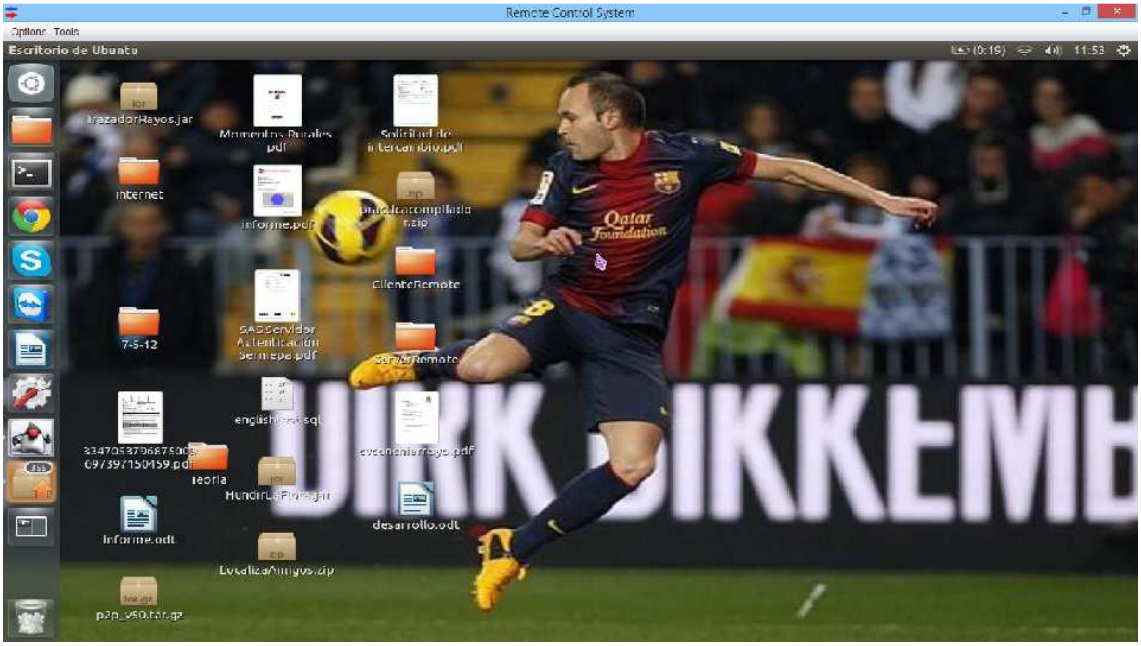
Nota: La interfaz de la beta de autenticación por reconocimiento facial no la he puesto. Esto es debido a que se trata de una serie de pruebas (bastante manuales) no integradas con el resto del software.

En la parte del cliente del software Remote Control tenemos el siguiente interfaz:

Podemos ver un menú superior con las pestañas *Options y Tools.*

Para establecer conexión, que es lo primero que se debe hacer, en la opción Options se debe establecer conexión e introducir la IP pertinente.

Una vez establecida la conexión con el servidor se habilitan en el resto de opciones e inicia el servicio.



En el momento que se ha establecido la conexión el programa (cliente) ya es capaz de capturar todos los eventos de teclado y ratón del cliente y enviarlos por la conexión para que se ejecuten en el servidor.

En *Tools* tenemos las opciones de:

- coger archivos del servidor remoto
- poner archivos del servidor remoto
- sacar una foto con la webcam del servidor y traerla al cliente
- manejar la calidad del servicio según el estado de la red
- mostrar/ocultar el interfaz remoto del servidor
- generar informe del estado de la conexión

En *Options* además de la ya comentada utilidad para establecer conexión tenemos las diferentes opciones de cerrar conexión:

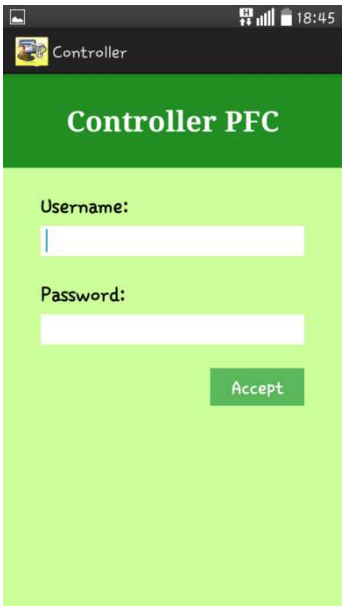
- Cerrar conexión: cierra la conexión con el servidor pero no cierra el programa.
- Cerrar cliente y conexión: cierra la conexión y termina el cliente.
- Cerrar cliente, servidor y conexión: cierra la conexión, termina el cliente y manda cerrarse al servidor.

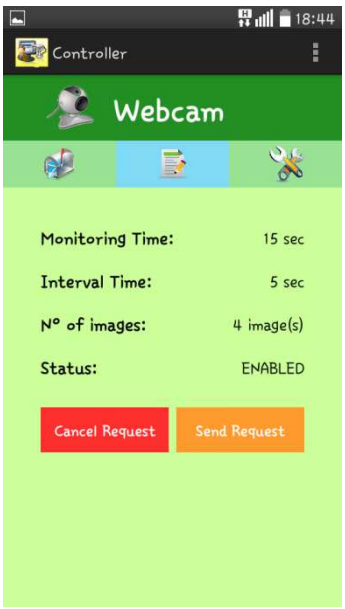
Por último, en el manual de usuario, voy a enseñar la aplicación para Android que actúa a modo de gestor del sistema (al igual que la web) pero especializada para este tipo de dispositivos.

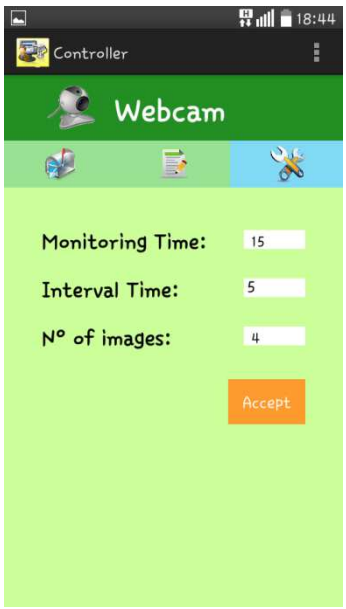
Su funcionamiento es muy intuitivo:

- Interfaz para la autenticación, mediante usuario y contraseña, para el primer acceso a la aplicación.
- Menú para seleccionar cada uno de los dispositivos.
- Bandeja de entrada, solicitud/cancelación de nuevas peticiones, modificación de parámetros dentro de la vista de cada uno de los dispositivos.
- Opciones para descargar, enviar por correo y eliminar cada uno de los recursos de los diferentes dispositivos.

A continuación mostraré alguna imagen de las pantallas que se pueden ver en la aplicación a modo de ejemplo.







CONTROL REMOTO DE LOS PRINCIPALES DISPOSITIVOS DE UN ORDENADOR MEDIANTE UN CLIENTE WEB

Realizado por: Javier Ladrón Arroyo

ÍNDICE

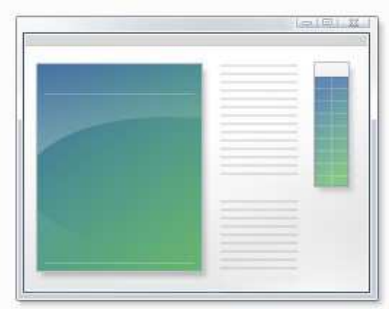
- Descripción y Objetivos
- Propuesta
- Análisis y Diseño
- Dificultades encontradas
- Soluciones
- Pruebas
- Presupuesto
- Conclusiones
- Líneas futuras

Descripción y Objetivos

- Monitorizar entorno
- Aprovechar recursos
- Gestión del entorno
- Técnicas IA aplicadas

PROPUESTA

Ordenador Personal + Sofware Controlador



Servidor Web + Base de Datos

Interfaces de gestión. Website + Android App

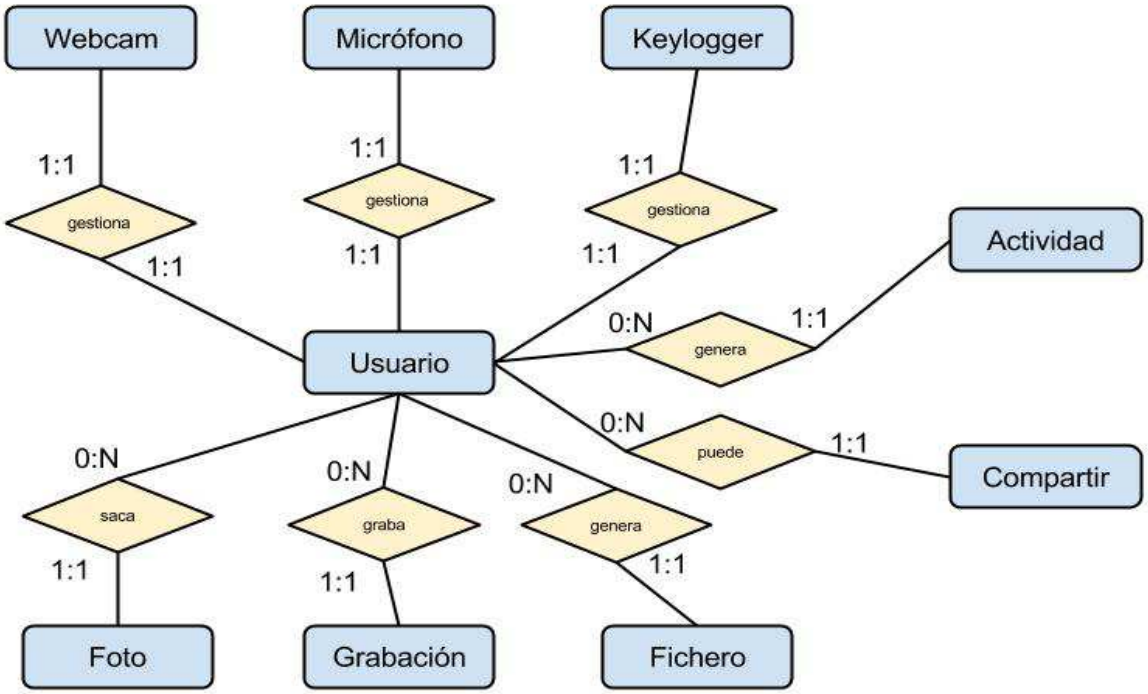
Propuesta. Anexo

- Software Control Remoto

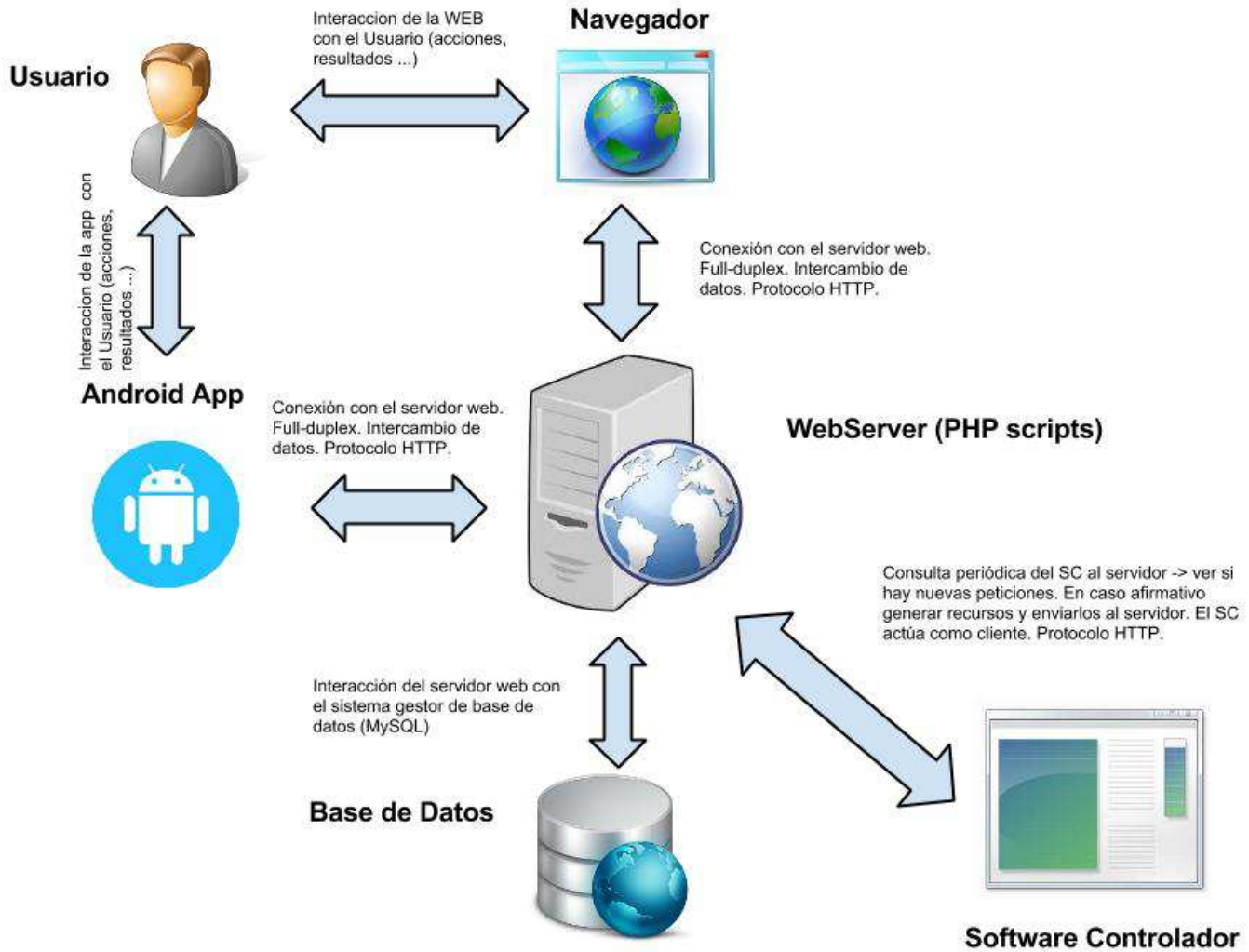
Cliente

Servidor

Análisis y diseño. Diagrama entidad relación



Análisis y diseño. Arquitectura del sistema



DIFICULTADES ENCONTRADAS

- Conocimiento nuevas tecnologías
- Utilidades adicionales
- Minimización de retardos y errores
- Analizador sintáctico integrado
- Reconocimiento facial
- Autentificación Software Controlador
- Restricciones locales y de red
- Integridad en la gestión de datos
- Prestaciones Software Control Remoto
- Interfaces web y de aplicación

SOLUCIONES

- Documentación oficial
- Búsqueda de librerías y herramientas
- Concurrencia y control de errores
- Gold Parser Builder
- OpenCV (Algoritmo PCA)
- ID único asociado a una cuenta
- Servidor Web y Base de Datos pública
- Servidor Web <-> Base de Datos
- Protocolo de conexión flexible y robusto
- Librerías externas y omisión de “pasos”

PRUEBAS REALIZADAS

UNITARIAS

- Autentificación sistema SC
- Analizador sintáctico
- Interacción correcta con el Servidor Web
- Captura, envío y recepción de eventos
- Generación correcta de informe
- Ejemplo reconocimiento facial
- Modificación de parámetros
- Envío y cancelación de recursos
- Acceso y gestión de recursos
- Pruebas de seguridad

PRUEBAS REALIZADAS

DE CARGA

- Subida masiva de recursos
- Redes LAN con poco ancho de banda
- Acceso a recursos de gran tamaño

PRESUPUESTO

Análisis y Diseño (376 h) -> 16920 €

Implementación (476 h) -> 18088 €

Pruebas (168 h) -> 6384 €

Documentación (112 h) -> 4256 €

+

Proyecto -> 45648 €

Coste hora análisis y diseño: 45€ | Coste hora desarrollo: 38€

Conclusiones y líneas futuras

- Origen idea del PFC
- Nuevas tecnologías desconocidas
- Servidor web dominio público
- Incorporación de IA en el proyecto

Conclusiones y líneas futuras

- Analizador sintáctico integrado
- Recurso vídeo
- Reconocimiento facial
- Protocolo software control remoto
- Dinamización de la interfaz web (AJAX)
- Encriptación de los protocolos de comunicaciones
- Aumentar utilidades interfaz aplicación