UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

PROYECTO FIN DE CARRERA

HERRAMIENTAS DE APOYO A LA EMISIÓN DE CLASES PRESENCIALES

Álvaro J. Rubio Redondo

Enero 2015

HERRAMIENTAS DE APOYO A LA EMISIÓN DE CLASES PRESENCIALES

AUTOR: Álvaro J. Rubio Redondo

TUTOR: Jesús Bescós Cano

Video Processing and Understanding Lab

Dpto. de Tecnología Electrónica y de las Comunicaciones

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Enero 2015

Palabras clave

Emisión de clases presenciales, *streaming* de vídeo y audio en directo, codificación, servidor, *framework*, aplicación web.

Resumen

Este proyecto tiene por objetivo tener un sistema de emisión en directo de clases vía *streaming* completamente funcional. Para ello se tendrá que estudiar la infraestructura con la que se cuenta y buscar las herramientas que se adapten mejor a esta infraestructura para cumplir el objetivo.

Además se tendrá que crear un sistema que controle el acceso al visionado del vídeo por parte de los distintos usuarios en función de los permisos con los que cuenten.

Dado que se pretende que el sistema se utilice realmente en el ámbito académico, será necesario acompañar a este documento de los manuales necesarios para la correcta administración del sistema.

Abstract

This Project aims to have a fully functional broadcast system via live streaming for classes. It will be necessary to study the infrastructure that is already installed and search for tools that best fit this infrastructure to meet the target.

Additionally it will be needed to create a system that controls access to video viewing by the users based on the permissions they have.

Since it is intended that the system be used in the academic field, it will be necessary to accompany this paper with the manuals for the proper administration of the system.

Agradecimientos

En primer lugar quiero agradecer a mi tutor, Jesús Bescós Cano, la oportunidad de realizar este proyecto. Gracias a su dedicación y sus sabios consejos ha sido posible la finalización del mismo.

También dar las gracias a José María Martínez, a Juan Carlos SanMiguel y al resto de compañeros del grupo VPULab por su ayuda y sus contribuciones al proyecto.

A los profesores y compañeros de la carrera. Sin los profesores no habría adquirido todos los conocimientos de los que ahora dispongo, pero sin los compañeros no habría podido asimilar todos esos conocimientos. En especial gracias a Carol, Almu y Gustavo. Los largos días haciendo prácticas no habrían sido lo mismo sin unos compañeros tan especiales.

A la música, que gracias a ella he podido conocer a una gente que de otra manera no habríamos coincidido. Gracias por estos años y por todos esos momentos irrepetibles sobre un escenario o en los ensayos. Los jueves tienen otro significado gracias a vosotros.

Gracias a Guille por su gran ayuda en la última fase del proyecto, y por supuesto gracias al resto de mis amigos de Tres Cantos. Me habría sido imposible llegar hasta aquí sin vosotros. Llevamos tantos años juntos que ya sois como mi segunda familia. Aunque la vida nos lleve por caminos distintos y nos separen miles de kilómetros, sé que siempre estaréis ahí y que cuando nos volvamos a juntar será como si no hubiese pasado el tiempo.

Agradecer muy especialmente a mis padres todo el apoyo que me han dado, su paciencia y su dedicación durante toda mi vida. A mi hermana por su capacidad de sacarme una sonrisa hasta en los peores momentos y por haber compartido junto a mis padres los mejores momentos de mi vida.

El camino ha sido largo y en ocasiones duro, pero todos habéis estado ahí cuando más os necesitaba. Tan sólo espero que en la nueva etapa que me toca vivir tenga tanta suerte como la que he tenido hasta ahora.

MUCHAS GRACIAS

Álvaro J. Rubio Redondo

Enero 2015

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

En este primer capítulo se realizará una descripción de la base de la idea de la que parte este proyecto, marcando unos objetivos y una estructuración de la memoria.

1.1 Motivación

Debido al gran avance de las tecnologías de la información en los últimos años podemos tener acceso a diferentes sistemas de información desde nuestra propia casa a través de Internet. A veces es simplemente una comodidad el tener esta opción de acceso, pero en otros casos se convierte en algo necesario para ciertas personas. Es por ello que se necesita tener un sistema de videoconferencia en las aulas, adaptado al tipo de cámara que se instala en ellas.

A través del sistema de videoconferencia instalado en las aulas podremos dar acceso a la sesión que se está impartiendo en ella a cualquier alumno, que por causas justificadas (no se quiere reducir la asistencia a las clases), no sea posible que asista a la sesión. También sería posible que en ponencias cualquier componente del tribunal que se encuentre demasiado lejos, pueda visualizarla y otorgar la nota correspondiente. Finalmente sería también una buena opción en conferencias dar acceso a ciertas personas acreditadas que tengan problemas para asistir.

Desgraciadamente, los servidores y el sistema de control de acceso que proveen las cámaras instaladas en las aulas son muy limitados y no ofrecen la libertad que se requiere para tener un sistema robusto que se pueda utilizar en el ámbito académico. Es por ello que, partiendo del sistema instalado y de las investigaciones realizadas en Proyectos anteriores [1] [2], hay que desarrollar un nuevo sistema que ofrezca todas las características deseadas.

1.2 Objetivos

El objeto de este proyecto es tener un sistema de emisión en directo de clases vía *streaming* completamente funcional. Se busca que el acceso al vídeo por parte de los usuarios sea lo más sencillo posible, por lo que se evitará el uso de aplicaciones específicas que los usuarios tengan que instalarse en sus ordenadores y se estudiará la manera de poder integrar la aplicación dentro de los navegadores web más comunes. Así mismo se pretende que el sistema pueda ser implantado en el futuro, por lo que es importante que la complejidad de su mantenimiento no sea muy elevada y que una persona con un conocimiento medio en tecnologías de la información, junto con este documento, pueda desempeñar las tareas de implantación y mantenimiento del sistema.

Para ello se divide el trabajo en varios objetivos:

1. Estudio de los antecedentes

   Antes de afrontar el proyecto hay que realizar una revisión de los antecedentes tecnológicos al sistema. En este caso, la investigación se centrará en sistemas de emisión de vídeo en directo a través de Internet.

2. Diseño

   Una vez estudiado el funcionamiento de los sistemas actuales, se estudiará la mejor arquitectura que resuelva el problema y se planteará el desarrollo de las aplicaciones necesarias.

3. Desarrollo

   Sabiendo el diseño del sistema que se quiere implementar, habrá que estudiar qué herramientas se adaptan mejor al sistema y comenzar a trabajar en su desarrollo.

4. Integración del sistema y pruebas finales

   Se combinarán todas las herramientas desarrolladas en los apartados anteriores y, una vez integradas, se procederá a realizar las pruebas del sistema completo para verificar su funcionalidad.

5. Evaluación de los resultados y trabajos futuros

   En este punto se valorará si se han cumplido los objetivos iniciales del proyecto y se plantearán posibles mejoras para futuros trabajos.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo 1:** Introducción, objetivos y motivación del proyecto.

- **Capítulo 2:** Estudio de los antecedentes y de los sistemas actuales.

- **Capítulo 3:** Descripción de la infraestructura instalada y diseño del sistema.

- **Capítulo 4:** Estudio de las herramientas que mejor se adaptan al diseño y desarrollo del sistema.

- **Capítulo 5:** Integración de las herramientas, pruebas y evaluación de los resultados obtenidos.

- **Capítulo 6:** Conclusiones y trabajo futuro.

2 Antecedentes

Este capítulo pretende dar una visión histórica de la tecnología *streaming* y mostrar algunos ejemplos de los sistemas de emisión en directo comerciales que existen en la actualidad.

2.1 Breve historia de la retransmisión por Internet

La emisión de un evento en directo a través de Internet lleva poco tiempo entre nosotros, ya que fue a partir de 1995 cuando empezaron a aparecer los primeros servicios de *streaming*. Hasta ese momento, la reproducción de contenido multimedia implicaba tener que descargar completamente el archivo al disco duro local. Con el lanzamiento de RealAudio 1.0 todo esto cambió y el usuario pudo consumir el producto en paralelo mientras se descargaba. Este tipo de tecnología funciona mediante un búfer de datos que va almacenando la información que se va descargando en la estación del usuario para luego mostrarle el material descargado.

El auge de plataformas como YouTube y el aumento del ancho de banda disponible para una gran parte de la población han supuesto que muchas cadenas de Televisión y otros servicios se lancen a emitir sus programas en directo. Desgraciadamente, al ser una tecnología en desarrollo, todavía no existe un estándar para la emisión en directo a través de Internet, y las empresas que dan este servicio son muy celosas con sus métodos y no proporcionan información.

2.2 Sistemas comerciales de emisión por Internet

A continuación se presentan algunos de los sistemas más populares de retransmisión vía *streaming* por Internet.

- **Wowza Streaming Engine:**

  Herramienta creada por Wowza Media Systems que hace muy sencilla la tarea de emitir archivos multimedia en directo o bajo demanda, ya que dispone de un *software* en el que se le pueden especificar las características del archivo o cámara IP, protocolos de envío, calidad del vídeo o *bitrate* deseado. Wowza también dispone de la posibilidad de incrustar el vídeo en la página web deseada mediante su propio sistema de reproducción. El servicio básico cuesta 55 dólares al mes por cada flujo de vídeo que se emita, existiendo también una versión de evaluación gratuita pero con características muy limitadas.

Figura 2-1: Logo Wowza

- **Ustream:**

  Elegida como la plataforma número 1 en emisión de vídeo en directo vía *streaming*. Además de ofrecer servicios de emisión y reproducción que ofrece Wowza, también ofrece *software* de edición y producción de vídeos. Tiene un rango muy grande de precios, dependiendo de los servicios que se desean contratar, y van desde los 99 hasta los 999 dólares.

Figura 2-2: Logo Ustream

- **Livestream:**

  Muy parecida a los dos sistemas anteriores, es de los sistemas más populares que se pueden encontrar, con más de 40 millones de reproducciones de eventos al mes. Su precio es de 42 dólares al mes sin opción de incrustar vídeos en una web y 199 dólares al mes con esta opción.

Figura 2-3: Logo Livestream

Los sistemas comerciales no son una opción para el sistema propuesto. La ventaja que tienen es que al ser grandes empresas disponen de varios servidores de gran potencia y pueden garantizar el funcionamiento del servicio. Sin embargo, el alto precio que hay que pagar por usar sus sistemas hace inviable su uso en este proyecto.

Por lo tanto, habrá que diseñar un sistema propio de retransmisión de vídeo a través de Internet. Para ello se diseñará una arquitectura que se adapte a la infraestructura de la que se dispone.

3 Especificaciones y diseño

En este capítulo se describirá la infraestructura disponible, se describirán los problemas que presenta y se diseñará la arquitectura del sistema que se desarrollará.

3.1 Descripción de la cámara

El modelo de cámara utilizado en este proyecto es una cámara PTZ, la SONY SNC RZ50P. La cámara se encuentra conectada a la red IP de la Escuela y permite el acceso desde cualquier PC conectado a Internet.

Figura 3-1: Cámara SONY SNC RZ50P

La cámara SONY SNC RZ50P es una cámara PTZ, que gracias a estar motorizada permite los movimientos en horizontal y vertical. Tiene integrado un *zoom* óptico de hasta 26x (ampliable mediante *zoom* digital), y alcanza una tasa de imágenes de 25 fps y una resolución máxima de 704x576 píxeles.

La cámara viene con un servidor por defecto al que es posible conectarse a través de una dirección IP. Si se hace desde un explorador de Internet nos lleva hasta el portal de la cámara:

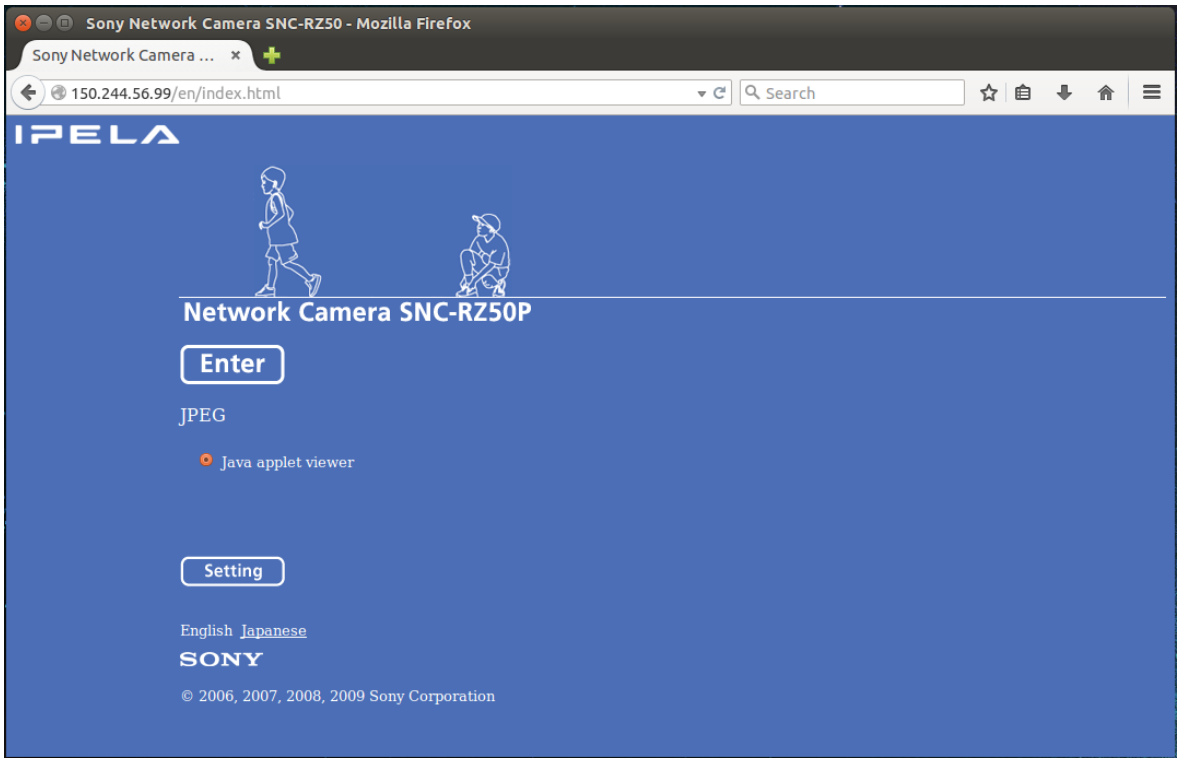

Figura 3-2: Portal de la cámara

Desde aquí es posible visualizar el vídeo de la cámara y configurar valores de la cámara o de su servidor, todo ello dependiendo del usuario que acceda y de sus permisos.

Este portal ya sería un sistema válido para emitir clases presenciales, pero las limitaciones que tiene impuestas por SONY, como por ejemplo el bajo número de conexiones permitidas al mismo tiempo, o una única configuración de vídeo hacen que este sistema sea demasiado básico.

Por otra parte, SONY provee de una serie de librerías escritas en C++ para crear aplicaciones de acceso y control de la cámara. Sin embargo estas librerías sólo se pueden utilizar en este tipo de cámaras, por lo que si se pretende desarrollar un sistema en el que se puedan implementar distintas configuraciones de cámaras, la utilización de estas librerías dejaría al sistema muy limitado.

3.2 Diseño de la arquitectura

Tras el estudio realizado en el punto 2.2 y el estudio de la infraestructura disponible, se ha decidido crear todo el sistema a partir de la integración de distintas herramientas disponibles de manera gratuita.

El esquema general de la arquitectura elegida es el siguiente:

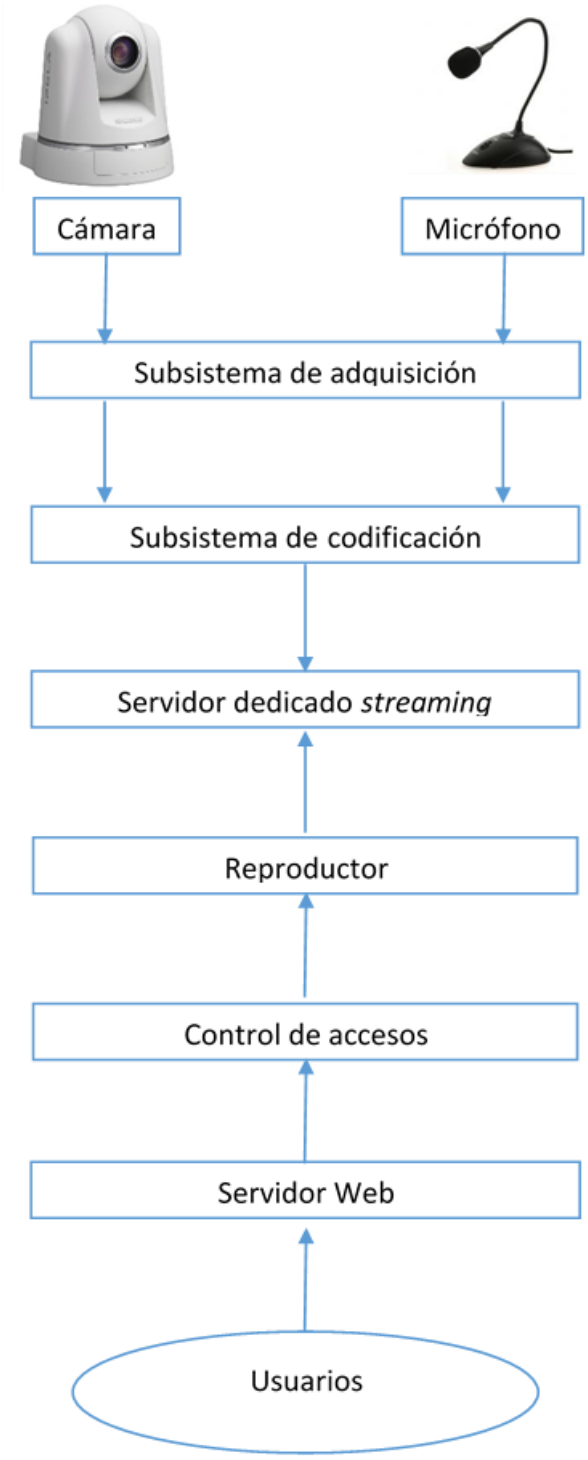

Figura 3-3: Arquitectura del sistema

- Para empezar es necesario crear un subsistema que obtenga los flujos de vídeo y audio desde el servidor de la cámara. Este servidor tiene separados estos dos

flujos, luego habrá que tenerlo en cuenta para encontrar la aplicación que pueda descargar ambos flujos en paralelo.

- Una vez obtenido el vídeo y el audio por separado, habrá que juntarlos y aplicarle una serie de transformaciones. Habrá que usar un formato apropiado para el *streaming* en directo y dar la posibilidad de retrasmitir vídeo con varias calidades diferentes.

- Cuando se disponga de un flujo único de vídeo y audio será necesario enviarlo a un servidor dedicado para alojar el vídeo en directo.

- Para evitar que los usuarios tengan que instalar aplicaciones en sus ordenadores personales, se creará una página web a la que los usuarios que deseen visualizar el vídeo tendrán que acceder. Esta página web tendrá que ir alojada en un servidor web que tenga acceso al servidor que aloja los flujos de vídeo. El acceso a la página web no puede ser global, por lo que habrá que crear una aplicación web que gestione el acceso de los distintos grupos de usuarios mediante la concesión de permisos. Los perfiles que se deben de manejar son los de alumnos, profesores y asistentes a una conferencia. Cada uno de estos perfiles debe de tener características únicas, como por ejemplo distinta calidad en la reproducción del vídeo.

4 Desarrollo

En este capítulo se va a presentar el proceso de desarrollo del sistema. Como se vio en el capítulo anterior, el sistema se divide en una serie de subsistemas, por lo que el desarrollo se hará de cada subsistema individualmente, para en el capítulo 5 integrar todas las partes y realizar las pruebas. La estructura del desarrollo se basa en la presentación de varias herramientas estudiadas y la elección de la herramienta que se va a implementar.

4.1 Sistema Operativo

La primera elección que hay que tomar es el Sistema Operativo para desarrollar el proyecto. Las opciones barajadas son Windows 8 o Ubuntu 12.

Dado que el sistema va a necesitar usar herramientas de servidores y el rendimiento del Sistema Operativo va a ser determinante, se va a utilizar Ubuntu por su mayor grado de personalización, facilidad a la hora de programar *scripts* y su facilidad para crear servidores.

Por lo tanto, la investigación de las herramientas necesarias para llevar a cabo el sistema se centró en la compatibilidad con Ubuntu 12.04. Por desgracia, en junio de 2014 cuando ya se había avanzado significativamente con el desarrollo del sistema, Ubuntu decidió dejar de dar soporte a la versión 12.04 de su Sistema Operativo para forzar la actualización a su sistema más moderno, el 14.04. Esto ocasionó que no se pudiesen instalar nuevas aplicaciones y que la compatibilidad de muchas de ellas se viera afectada en el nuevo sistema. Finalmente, tras un retraso de varios meses configurando las aplicaciones, se consiguió hacerlas funcionar en el nuevo Sistema Operativo. En los anexos se ha creado una guía de instalación, en Ubuntu 14.04, de todas las herramientas que se han usado en el desarrollo del sistema.

4.2 Subsistema de adquisición

Las cámaras instaladas tienen su propio servidor en el que alojan el flujo de vídeo y audio, al que se accede mediante una URL con la dirección IP de la cámara. Ambos flujos están en secciones separadas, el vídeo en formato *mjpeg* y el audio codificado con el estándar G.711. Para acceder a este servidor es necesario autentificarse con un usuario y una contraseña y hay que tener en cuenta que la descarga desde el servidor se tiene que ejecutar de manera continua y que ese flujo habrá que redireccionarlo a otra aplicación para su codificación. A continuación se presentan varias aplicaciones que se han estudiado:

4.2.1 cURL

Figura 4-1: Logo cURL

cURL es una herramienta de código abierto y gratuita para usar en un intérprete de comandos para transferir archivos con sintaxis URL. El principal propósito y uso para cURL es automatizar transferencias de archivos o secuencias de operaciones no supervisadas. Soporta multitud de protocolos de envío de archivos. Además permite la descarga de archivos con autentificación de usuario y contraseña y la redirección de la salida mediante *pipes.*

Por el contrario, no permite la descarga en paralelo de varios archivos desde una misma instrucción.

4.2.2 LFTP

Esta herramienta gratuita también permite la transferencia de archivos mediante varios protocolos, sin embargo esta sí permite la descarga de archivos en paralelo.

Figura 4-2: Logo LFTP

Está especialmente orientada a la creación de copias de seguridad para servidores FTP, pero también soporta protocolos como FTPS, HTTP, HTTPS y SFTP entre otros.

Se utiliza en el intérprete de comandos, pero mediante su propia *shell.*

4.2.3 ARIA2C

Aria2C es otra herramienta gratuita para la transferencia de archivos con soporte para protocolos como HTTP(S), FTP, BitTorrent y Metalink. Como la herramienta LFTP, también admite la descarga en paralelo, pero una de sus características es que no requiere mucho uso de memoria ni de tiempo de CPU. Dado que nuestro sistema va a requerir grandes consumos de procesador, esta es una característica que hay que tener en cuenta. Sin embargo, no es tan fácil redirigir la salida como con cURL.

4.2.4 Elección de herramienta de adquisición

En principio se buscaba una aplicación que permitiese la descarga paralela, ya que el vídeo y el audio se encuentran en secciones separadas en el servidor. Por lo tanto se hicieron pruebas con LFTP y con ARIA2C.

Utilizando LFTP no fue posible conectarse al servidor para obtener el flujo de vídeo ya que la aplicación no implementa la opción de acceso con usuario y contraseña.

Con ARIA2C se llegó a descargar ambos flujos de manera paralela, pero después no se podía redireccionar la salida como se pretendía en un primer momento. Se intentó cambiar de estrategia, y en lugar de redireccionar la salida crear un archivo que luego leyese el subsistema de codificación. Este método no fue válido y se descartó.

Finalmente se optó por utilizar **cURL**, herramienta más sencilla que no disponía por defecto de la descarga paralela. Sin embargo, mediante la creación de un *script* se solventó ese problema. En el apartado 0 se explicará cómo se ha implementado este subsistema junto con el subsistema de codificación, pero un ejemplo de la sintaxis que utiliza cURL puede ser el siguiente:

```
curl -u usuario:contraseña IP_SERVIDOR | appCodificación
```

- Utilizado desde una terminal, el comando `curl` ejecuta la aplicación.

- El argumento `-u` es optativo, y le dice a la aplicación que el servidor requiere de un usuario y una contraseña para su acceso. A continuación del argumento se introduce el usuario y la contraseña separados por dos puntos (:).

- El último argumento necesario es la dirección IP de la que tiene que descargar.

- Como no queremos crear un archivo, sino ejecutar una aplicación que le entre directamente el flujo, la terminal de Linux dispone del comando `|` que crea una tubería que redirecciona la salida estándar de la aplicación a su izquierda a la entrada estándar de la aplicación a su derecha.

4.3 Subsistema de codificación

Una vez obtenido el flujo de audio y vídeo es necesario aplicarle una serie de transformaciones. Primero hay que estudiar los distintos formatos disponibles para hacer *streaming*, y posteriormente estudiar las herramientas disponibles para transformar al formato elegido.

4.3.1 Formatos de vídeo más populares

4.3.1.1 AVI

Desarrollado por Microsoft en 1992 es un formato contenedor multimedia. Esto quiere decir que admite varios formatos de audio y vídeo. Es el formato estándar para almacenar vídeo digital porque puede contener vídeo de una calidad excelente. Sin embargo, el peso del archivo resulta siempre muy elevado, por lo que no es un formato recomendado para publicarlo en Internet.

4.3.1.2 H.264 o MPEG-4

Es una norma que define un códec de vídeo de alta compresión, desarrollada conjuntamente por el ITU-T Video Coding Experts Group (VCEG) y el ISO/IEC Moving Picture Experts Group (MPEG). La intención del proyecto H.264/AVC fue la de crear un estándar capaz de proporcionar una buena calidad de imagen con tasas binarias notablemente inferiores a los estándares previos (MPEG-2, H.263 o MPEG-4 parte 2), además de no incrementar la complejidad de su diseño.

La mayoría de los dispositivos portátiles y computadoras reproducen este formato sin necesidad de complementos adicionales, y además se puede crear un archivo MP4 con casi cualquier software de edición de video.

4.3.1.3 FLV

FLV o Flash Video es un formato contenedor propietario usado para transmitir video por Internet usando Adobe Flash Player. Utiliza el códec Sorenson Spark y el códec On2 VP6, con los que se puede conseguir una alta calidad visual con *bitrates* reducidos. Por otro lado es computacionalmente más complejo y por lo tanto puede tener problemas al utilizarse en sistemas con configuraciones antiguas. El soporte para codificar el archivo FLV es proporcionado por una herramienta de codificación incluida en Macromedia Flash 8 Professional de Adobe, las herramientas de codificación Flix de On2, Sorenson Squeeze, FFmpeg y otras herramientas de terceros.

Repositorios de vídeo tan conocidos como YouTube o Google Video utilizan este formato, y teniendo en cuenta su popularidad, el complemento Flash normalmente es uno de los primeros que los usuarios instalan al configurar sus navegadores. Además la posibilidad de usarlo para *streaming* a través del protocolo RTMP hace de este formato un gran candidato para nuestro sistema.

4.3.1.4 Elección del formato

Una vez vistos los diferentes formatos existentes, el que más se adecúa a las necesidades del proyecto es el formato **FLV**, debido a su gran expansión en aplicaciones de *streaming*. Habrá que asegurarse que la herramienta de codificación que se va a utilizar dispone de la capacidad para utilizar este formato y que la herramienta de reproducción sea compatible.

4.3.2 Herramientas de codificación

4.3.2.1 GStreamer

GStreamer es una biblioteca escrita en el lenguaje de programación C que sirve para desarrollar aplicaciones dedicadas al manejo de distintos medios [3].

Figura 4-3: Logo GStreamer

Las aplicaciones que nos permite crear van desde aplicaciones audiovisuales para la reproducción de audio y vídeo hasta aplicaciones más complejas para mezclar audio y vídeo, codificar e incluso realizar un *streaming* de los mismos.

Se trata de una herramienta muy potente, pero su elevada complejidad hacen de ella bastante inaccesible para los tiempos manejados en este proyecto.

4.3.2.2 Adobe Flash Media Live Encoder

Es un codificador multimedia que transmite audio y vídeo en tiempo real. Se trata de un producto de Adobe Systems pero sólo está disponible para Windows y Mac OS.

Utiliza el protocolo RTMP para servir el vídeo a los clientes a través de otra herramienta llamada Flash Media Server, también propiedad de Adobe. Además provee de una interfaz gráfica que hace muy intuitiva su configuración:

Figura 4-4: Interfaz Flash Media Live Encoder

Desgraciadamente, las herramientas de Adobe no son gratuitas y únicamente dejan un periodo de prueba limitado. De todas formas el estudio de esta herramienta ha proporcionado un conocimiento muy valioso del procedimiento para tener un *streaming* de vídeo en directo, por lo que se va a intentar emular estas herramientas por métodos de código abierto.

4.3.2.3 FFmpeg

Figura 4-5: Logo FFmpeg

FFmpeg es una colección de software libre que puede grabar, convertir y hacer *streaming* de audio y vídeo [4]. Incluye libavcodec, una biblioteca con multitud de códecs para codificar y decodificar vídeo y audio. FFmpeg está desarrollado en GNU/Linux, pero puede ser compilado en la mayoría de los sistemas operativos, incluyendo Windows. Es una herramienta muy potente pero sencilla de utilizar. Se puede usar en la terminal y a través de comandos se le van añadiendo opciones. Este método hace muy fácil comenzar a utilizarla y posteriormente mejorar el sistema conforme se va aprendiendo su uso.

4.3.2.4 Elección de la herramienta de codificación

La elección final fue utilizar **FFmpeg**. Se encontró abundante documentación y según las pruebas efectuadas se integraba bien con el sistema. En el A se explica cómo instalar esta herramienta en el ordenador y se detallan algunos comandos importantes para su uso. En el capítulo 5 se detalla la integración de las herramientas de codificación y de adquisición, pero un ejemplo de su utilización en el sistema propuesto sería el siguiente:

```
appAdquisicion | ffmpeg -y -an -f mjpeg -i - -s 320x240 -r 25 -b:v
                 548k -f flv "URLServidorDedicado"
```

- Con `ffmpeg` se ejecuta la herramienta.

- FFmpeg tiene multitud de argumentos optativos. Primero se configuran las opciones para la lectura del archivo entrante.

- `-i` indica la entrada. Al usar la entrada estándar y no un archivo o una dirección de red, se le indica con un guion.

- Posteriormente se configura el formato de salida, indicando tamaño, número de *frames* por segundo, *bitrate*, etc.

- Por último se le indica el lugar donde se quiere enviar la salida. Puede ser tanto la creación de un archivo en el disco duro local como enviarlo a un servidor a través de una URL.

4.4 Servidor dedicado

La razón de implementar un servidor dedicado es la de disponer de un lugar en el que, una vez obtenidos y transformados los flujos de vídeo y audio, éstos sean accesibles por los usuarios. La elección del tipo de servidor tendrá que ir ligada a la elección del protocolo de transmisión que se va a utilizar. A continuación se presentan algunos de los servidores estudiados y los protocolos que usan:

4.4.1 FFserver – RTP

Parte del paquete FFmpeg, es un servidor para audio y video compatible con la emisión en directo, que trabaja al mismo tiempo que la codificación del archivo multimedia. Se configura a través de un archivo de configuración en el que se le especifican los flujos multimedia entrantes, se le configura la codificación que se les quiere dar y por último se crean los nuevos flujos o *feeds* que serán a los que se conectarán los clientes para visualizar el video.

FFserver actúa como servidor HTTP para recibir los flujos y como servidor RTP o HTTP para servirlos.

Como ventaja tiene que es muy sencillo de utilizar, ya que tan sólo hay que configurarle los puertos por los que recibe el flujo y el máximo de clientes que puede aceptar. Por contra, resulta bastante más complicada la tarea de publicar esos *feeds*, ya que los protocolos que utiliza no son de un uso muy extendido entre los reproductores.

4.4.2 Apache - HTTP Live Streaming

Apache es un servidor web HTTP de código abierto, y es uno de los servidores más conocidos y utilizados en Linux [5].

HTTP Live Streaming o HLS [6], es un protocolo de transmisión de archivos multimedia creado por la compañía Apple Inc. basado en el protocolo HTTP. Su funcionamiento se basa en el troceado de la secuencia en pequeños paquetes basados en HTTP, y de cada uno de estos paquetes se crean copias con distintas calidades. Cuando el

cliente está reproduciendo el archivo puede escoger entre multitud de descargas alternativas que contienen el mismo material pero con diferentes *bitrates*, haciendo que la reproducción sea adaptativa a las condiciones de la red.

La ventaja de este protocolo es que se puede utilizar en cualquier servidor HTTP. Sin embargo, es un protocolo que no lleva demasiado tiempo utilizándose y todavía no está desarrollado del todo, por lo que no existe un procedimiento estándar para su uso y no todos los navegadores soportan este protocolo.

De cara al futuro, es una opción a tener en cuenta, sobretodo porque es el único método disponible para poder reproducir vídeo en dispositivos portátiles de la compañía Apple.

4.4.3 Red5 Media Server – RTMP

Figura 4-6: logo Red5 Media Server

Red5 es un servidor multimedia de código abierto implementado en Java [7]. Tiene características similares al Adobe Flash Media Server, por lo que es perfecto para el formato FLV, que como ya se explicó en el punto 4.3.1.3 es un formato de gran importancia en el *streaming* multimedia. Tiene como ventaja que cuenta con módulos ya creados para la emisión y reproducción de archivos de vídeo, así como aplicaciones de chat y videoconferencia, por lo que su uso es bastante sencillo.

Real Time Messaging Protocol (RTMP) es un protocolo propiedad de Adobe, pero ésta lanzó una versión de uso público a mediados de 2009 [8]. La principal motivación de la creación de este protocolo fue para reproducir vídeo Flash.

RTMP está basado en TCP y mantiene una conexión constante entre el cliente y el servidor, lo que provoca una baja latencia en la comunicación. Comparado con el protocolo HTTP, la conexión constante de cada cliente con el servidor hace que no se pierda ningún dato en el envío, haciendo la comunicación más fluida y segura.

4.4.4 Elección del servidor dedicado

Las primeras pruebas se hicieron con FFserver para comprobar que el subsistema de adquisición y la herramienta de codificación funcionaban como se quería. Este servidor dio buenos resultados, pero su objetivo es sólo para pruebas y no para un uso más comercial.

Por lo tanto la elección del servidor se centró en la elección del protocolo. Tanto el HTTP Live Streaming (HLS) como el RTMP tienen ventajas e inconvenientes. En la siguiente tabla se muestran algunas diferencias:

HLS vs. RTMP		
	HLS	**RTMP**
Seguridad/Protección IP	Básica	Mejorada
Interactividad del reproductor	Básica	Amplia
Soporte Multicast	No aplica	Soportado
Penetración del protocolo	Emergente	Amplia
Firewall	Sin restricciones	Algunas restricciones
Soporte Bitrate variable	Sin impacto	Susceptible a picos de datos

Tabla 4-1: Comparativa HLS con RTMP

Aunque el HTTP Live Streaming es un protocolo que probablemente se acabe imponiendo en el futuro debido a que cuenta con muchas ventajas, en el momento de hacer este proyecto no se contaba con el suficiente soporte por parte de Apple ni de la comunidad como para implementar este protocolo. Por otra parte, las decisiones tomadas en los subsistemas anteriores hacen que la implementación del protocolo RTMP sea mucho más sencilla. La decisión por lo tanto fue la de utilizar el protocolo **RTMP** junto con el servidor **Red5**, que además cuenta con una aplicación llamada OflaDemo para retransmitir vídeo o audio en directo. Las instrucciones para la instalación de este servidor se encuentran en el Anexo E.

4.5 Reproductores

Existen multitud de reproductores de vídeo *streaming*, pero la investigación se va a centrar en los reproductores con capacidad de ser integrados en una página web.

4.5.1 Adobe Flash Player

Una vez más, Adobe provee de un reproductor que se adapta perfectamente al *streaming* de archivos multimedia. En este caso hace falta un editor, también proporcionado por Adobe, con el que habría que crear un archivo SWF donde se le especifica la información del tipo de flujo y del lugar del servidor donde se encuentra el vídeo. Posteriormente este archivo se coloca en el servidor que aloja la página web y a través de un objeto HTML se integrará dentro de la web.

Este reproductor es uno de los más usados, pero el editor vuelve a tener un límite de tiempo para su uso gratuito. Otra desventaja es que cada vez que se quiera cambiar alguna ubicación del vídeo en el servidor u otras configuraciones, hay que volver a editar el archivo SWF, lo que hace bastante engorroso su uso.

4.5.2 JW Player

Es un reproductor web de vídeo y audio de código abierto. Su gran ventaja es que puede reproducir tanto si el cliente tiene Flash instalado, como en HTML5, en gran auge debido a los dispositivos Apple.

Figura 4-7: Logo JW Player

Su uso es muy sencillo, ya que tan sólo hay que insertar un *sript* con la configuración del vídeo en el código HTML de la página web. Además se le pueden agregar distintas condiciones para que reproduzca un formato u otro dependiendo de la configuración que el usuario tenga en su dispositivo.

4.5.3 Elección del reproductor

En un principio se escogió el reproductor Adobe Flash Player para realizar las pruebas junto con el servidor FFserver. Se instaló la aplicación con bastantes problemas ya que no se encuentra disponible para Sistemas Operativos de Linux, pero se solventó gracias a herramientas de compatibilidad. Se creó un archivo SWF y se integró en una página web de prueba para evaluar el funcionamiento. Las pruebas dieron resultados aceptables, pero el hecho de tener que modificar el archivo SWF cada vez que se necesitaba agregar o cambiar algo en el servidor hizo que se abandonara este reproductor.

La elección tomada fue utilizar **JW Player**, uno de los reproductores más extendidos en la actualidad, cuya compatibilidad con el protocolo RTMP encajaba perfectamente con las herramientas elegidas anteriormente en este proyecto. Un ejemplo de inserción dentro del código HTML de la web es el siguiente:

```
<script type='text/javascript'>
    jwplayer('playerVZcSuaEnViqO').setup({
        file: 'URLservidorRTMP',
        width: '100%',
        aspectratio: '4:3',
        primary: 'flash',
        controls: 'false'
        rtmp: {
            bufferlength: 0.1
        },
    });
</script>
```

4.6 Control de accesos

Para garantizar el acceso de los usuarios a la reproducción del vídeo hace falta crear una aplicación que se tiene que integrar en una página web. Para ello se van a estudiar las distintas herramientas que pueden ayudar a la realización de tal fin, también llamadas *frameworks*, que van directamente ligadas a un lenguaje de programación. A continuación se van a explicar algunas de estas herramientas con sus respectivos lenguajes:

4.6.1 PHP

Figura 4-8: Logo PHP

Es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web de contenido dinámico. Fue creado 1995 y se convirtió en uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. PHP puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. El lenguaje PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores, entre los que se encuentran portales tan importantes como Wikipedia o Facebook. Sin embargo, el enorme número de sitios en PHP ha visto reducida su cantidad a favor de otros nuevos lenguajes no tan poderosos pero más cómodos a la hora de programar.

4.6.2 Python – Django

Figura 4-9: Logo Python

Python es un lenguaje multiparadigma, es un lenguaje interpretado, usa tipado dinámico y es multiplataforma.

Fue creado a finales de los años ochenta por y su filosofía hace hincapié en una sintaxis que favorezca un código legible. Incluso tiene unos principios descritos por un famoso desarrollador de Python, Tim Peters, entre los que destacan:

- Bello es mejor que feo.
- Explícito es mejor que implícito.
- Simple es mejor que complejo.
- Complejo es mejor que complicado.
- Plano es mejor que anidado.
- Disperso es mejor que denso.
- La legibilidad cuenta.
- Frente a la ambigüedad, rechaza la tentación de adivinar.
- Debería haber una -y preferiblemente sólo una- manera obvia de hacerlo.
- Si la implementación es difícil de explicar, es una mala idea.

Una de sus características es el uso de palabras donde otros lenguajes utilizarían símbolos. Por ejemplo, los operadores lógicos `!`, `||` y `&&` en Python se escriben `not`, `or` y `and`, respectivamente. El contenido de los bloques de código (bucles, funciones, clases, etc.) es delimitado mediante espacios o tabuladores, conocidos como indentación, antes de cada línea de órdenes pertenecientes al bloque. Python se diferencia así de otros lenguajes de programación que mantienen como costumbre declarar los bloques mediante un conjunto de caracteres, normalmente entre llaves `{}`.

Esta filosofía de hacer un código legible es muy útil para este proyecto. El tiempo de aprendizaje del lenguaje es corto, se pueden usar módulos ya desarrollados por la comunidad y entenderlos sin mucha dificultad y de cara a futuras mejoras en el proyecto es fácil que otra persona desarrolle sobre lo que ya está hecho.

Para facilitar el trabajo de crear una página web, existe un *framework* llamado Django. La meta fundamental de Django es facilitar la creación de sitios web complejos.

Django pone énfasis en el re-uso, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio *No te repitas* (DRY, del inglés Don't Repeat Yourself). Este principio significa que las definiciones deberían hacerse una sola vez, y si son usadas en distintos archivos es Django el que debe averiguar por sí mismo los nombres de, por ejemplo, las clases, en lugar de volver a definirlas. Python es usado en todas las partes del *framework*, incluso en configuraciones, archivos, y en los modelos de datos.

Figura 4-10: Logo Django

Lo que hace Django es automatizar muchas tareas de la programación web y así ahorrar tiempo al programador. Por ejemplo, al usar bases de datos, el programador sólo tiene que especificar el tipo de base de datos que se va a usar y Django se encarga de transformar el código Python desarrollado al lenguaje propio de la base de datos.

El uso de Python junto con Django está teniendo una gran aceptación en la comunidad de programadores, y cada vez son más las páginas web desarrolladas es este lenguaje. Algunos famosos portales que utilizan Django son Google, Yahoo y YouTube.

4.6.3 Ruby – Ruby on Rails

Figura 4-11: Logo Ruby

Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos. Combina una sintaxis inspirada en Python y Perl con características de programación orientada a objetos similares a Smalltalk.

Ruby está diseñado para la productividad y la diversión del desarrollador, siguiendo los principios de una buena interfaz de usuario. Su creador sostiene que el diseño de sistemas necesita enfatizar las necesidades humanas más que las de la máquina

Como ya se ha dicho, Ruby es orientado a objetos: todos los tipos de datos son un objeto, incluidas las clases y tipos que otros lenguajes definen como primitivas, (como enteros y booleanos). Toda función es un método, y las variables siempre son referencias a objetos, no los objetos mismos.

Tiene como desventajas que su ejecución es algo más lenta que PHP o Python y que el tiempo de aprendizaje es mayor, sin embargo cuenta con una gran comunidad con mucha actividad.

Ruby on Rails, también conocido como RoR o Rails, es un *framework* de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby, siguiendo el paradigma de la arquitectura Modelo Vista Controlador (MVC). Trata de combinar la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que con otros *frameworks* y con un mínimo de configuración.

Figura 4-12: Logo Ruby on Rails

Los principios fundamentales de Ruby on Rails incluyen *No te repitas* (ya visto en el caso de Django en el punto 4.6.2) y *Convención sobre Configuración*. Este último significa que el programador sólo necesita definir aquella configuración que no es convencional.

4.6.4 Elección del *framework*

Una vez analizadas las opciones, se decidió descartar el lenguaje PHP porque no es recomendable para la creación de aplicaciones complejas como las que se van a desarrollar en este proyecto.

Una vez centrada la búsqueda entre Django y Ruby on Rails, y por lo tanto entre Python y Ruby, la decisión se decantó hacia el lenguaje **Python** y el *framework* **Django**.

Ambos lenguajes gozan de una amplia comunidad de usuarios y de abundante documentación. El conocimiento de ambos lenguajes tiene mucha demanda en el mercado laboral y en el futuro seguirá aumentando, y ambos *frameworks* son suficientemente potentes y se adaptan perfectamente a las necesidades actuales y futuras del sistema. Python se adapta más a la mentalidad del programador y sus principios son compartidos por éste. Además, al ser un lenguaje que favorece la creación de un código legible se pensó que sería más sencillo para un futuro estudiante que desee continuar con este proyecto, independientemente de si conoce o no el lenguaje.

4.6.5 Desarrollo de la Aplicación Web

La instalación de Django y algunos apuntes sobre su funcionamiento se detalla en el Anexo C.

El desarrollo comenzó con el aprendizaje del lenguaje Python. No resultó muy difícil ya que hay disponible una gran cantidad de libros y tutoriales [9]. La sintaxis de Python es muy parecida a la de C, pero con una filosofía orientada a objetos, por lo que también se parece a Java.

Una vez aprendido el lenguaje, se comenzó a aprender el funcionamiento de Django. Gracias a la documentación consultada, tanto física como digital [10] [11], se llegó al suficiente conocimiento de este *framework* como para realizar el sistema propuesto.

El proceso de desarrollo seguido, así como una guía para futuros desarrollos con Django, se ha detallado en el Anexo D. A continuación se hará un resumen de la funcionalidad implementada:

- Se ha creado una base de datos en la que se registrarán los usuarios y que contendrá las distintas asignaturas o conferencias que se quieran retrasmitir. Todas estas entradas contarán con una serie de campos que el administrador del sistema tendrá que rellenar, como por ejemplo el nombre de la asignatura, el curso, la página web, etc.

- Cada asignatura o conferencia tendrá referenciado un número de usuarios (desde ninguno a cuantos se quiera) en los roles de alumnos y profesores (para asignaturas), o asistentes y conferenciantes (para conferencias). La asignación de estos usuarios determinará su permiso de acceso a la asignatura.

- Para el control de acceso se han creado tres grupos de usuario: Alumnos, Profesores y Conferencia. Es tarea del administrador registrar a cada usuario en un grupo, de lo contrario el usuario no podrá acceder al sistema.

- Una vez que el usuario tiene grupo asignado, podrá acceder a la sección privada del sistema. En esta página se mostrarán las asignaturas que el usuario tiene registradas, así como su horario y página web.
  Si un usuario se conecta al sistema en el horario estipulado de una asignatura que éste tiene registrada, podrá acceder a la página de reproducción del vídeo.

A continuación se muestran algunas de las plantillas implementadas para mostrar la funcionalidad de la web.

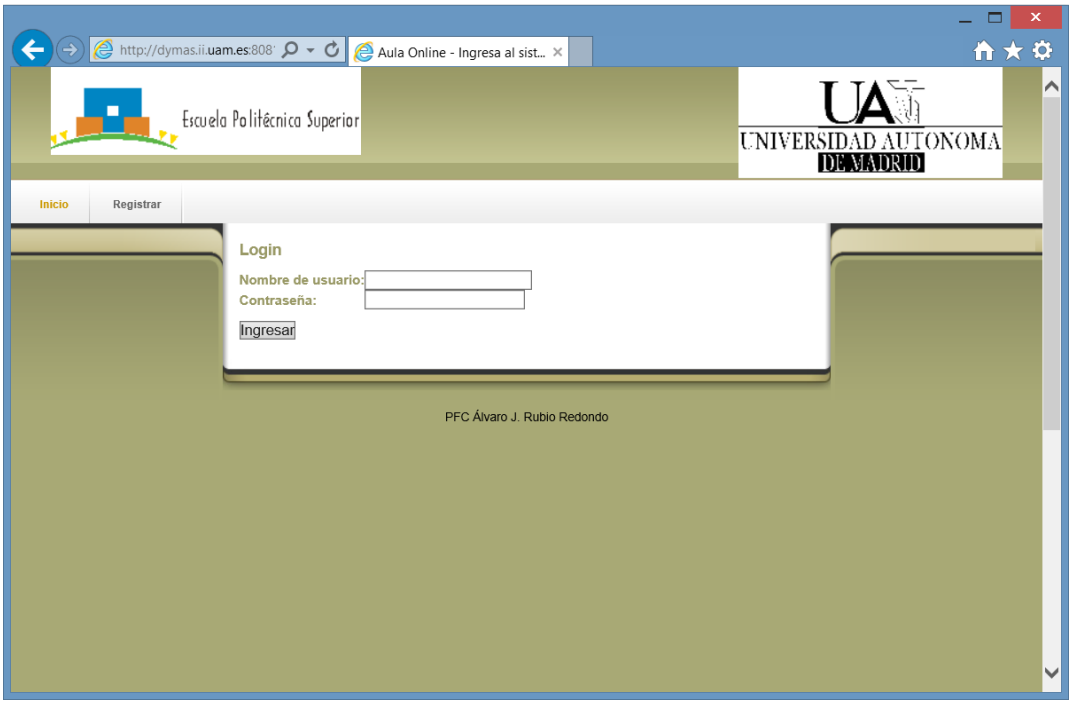

Figura 4-13: Plantilla de Inicio

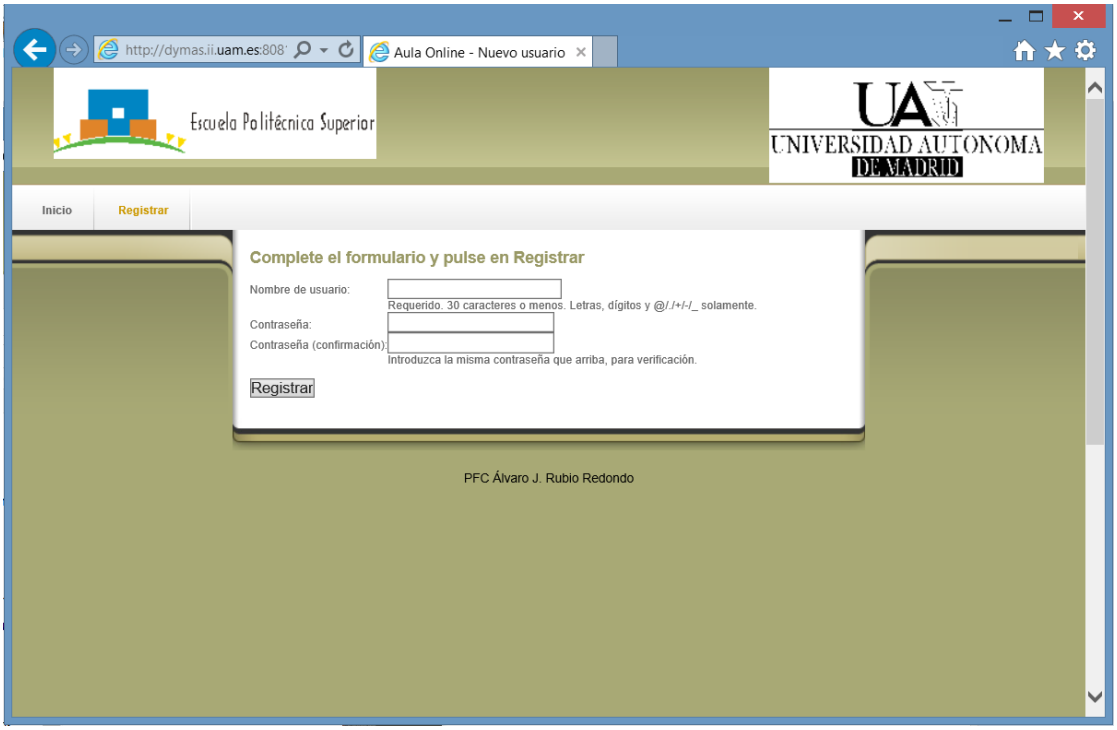

Figura 4-14: Plantilla Registro

Si el usuario se ha registrado e intenta acceder al sistema, pero aún no tiene un grupo asignado por el administrador, la página que verá será como la siguiente:

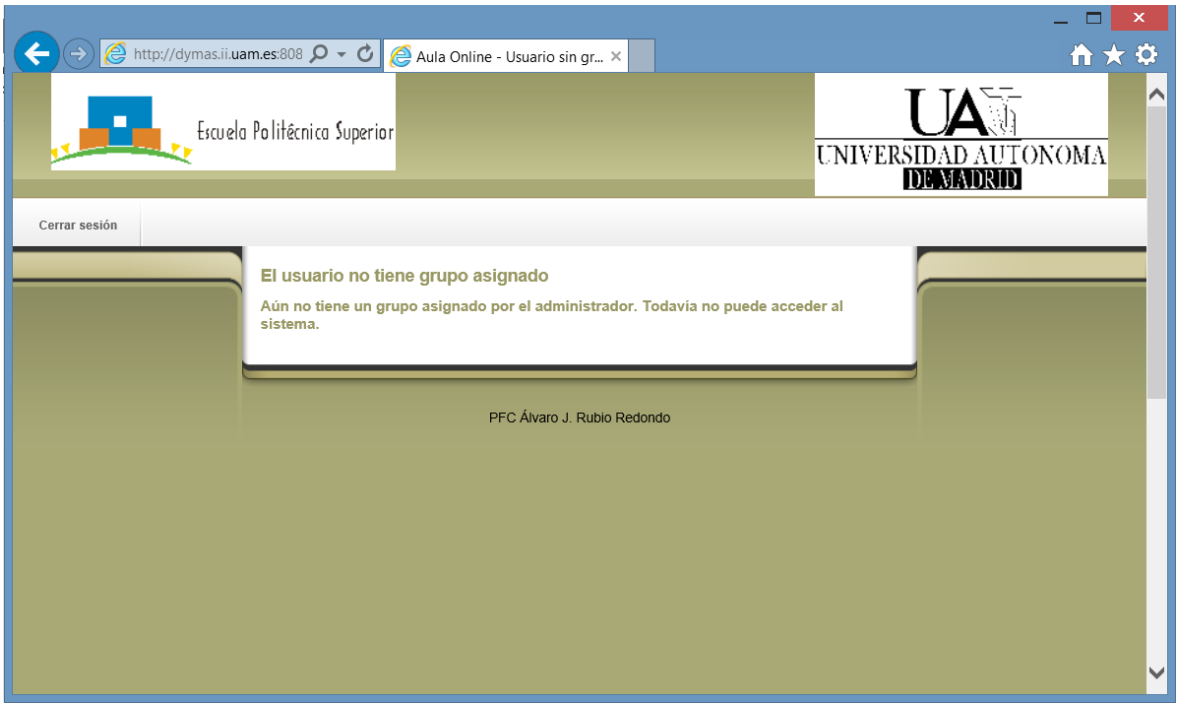

Figura 4-15: Plantilla Usuario sin grupo

Una vez que el usuario tiene un grupo asignado podrá entrar en el sistema y ver una página como la siguiente:

Figura 4-16: Plantilla Privado Alumno

En este caso se trata de un usuario del grupo Alumnos. En la columna central se muestra una lista con las asignaturas registradas para este usuario, con el horario de cada una de ellas y un hipervínculo a la web de la asignatura (en caso de que tenga). En la columna izquierda se muestra un mensaje de bienvenida con la última vez que el usuario entró en el sistema. En la barra horizontal se muestran las opciones de volver al Inicio, entrar en el Aula para visualizar la clase o cerrar sesión. La columna derecha se ha utilizado para poner unas sencillas instrucciones de utilización de la página.

Todas las columnas son editables, por lo que se puede añadir más información, relacionada con las asignaturas, noticias de la universidad, mensajes privados, etc.

Si el alumno entra en la pestaña de Aula dentro del horario estipulado para alguna de sus asignaturas registradas se le mostrará la siguiente página:

Figura 4-17: Plantilla Aula Alumno

Si el usuario es del grupo de Profesores, no tiene sentido mostrarle el vídeo de su propia clase. Por lo tanto se ha escogido una configuración de plantilla como la siguiente:

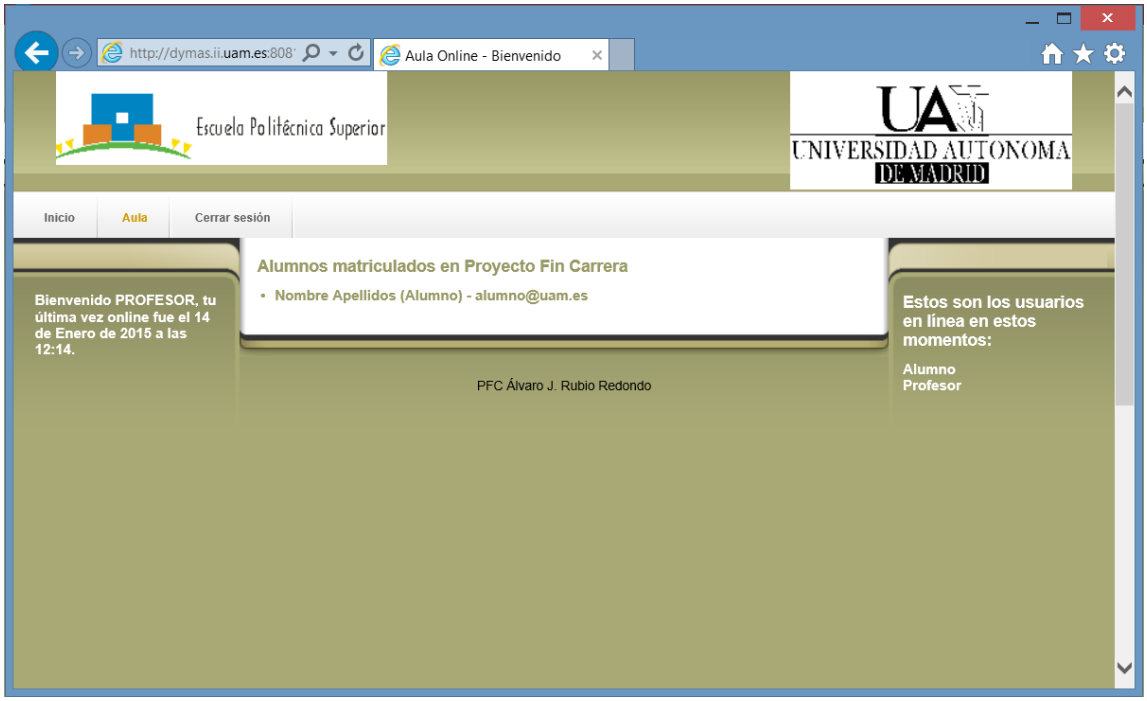

Figura 4-18: Plantilla Aula Profesor

La barra horizontal y la columna izquierda se mantienen, pero en la columna central se muestra una lista de los usuarios registrados en la asignatura que se está impartiendo en ese momento. La columna de la derecha muestra los usuarios que se encuentran conectado en el sistema en estos momentos.

Gracias a que Django es un *framework* ampliamente conocido en el entorno del diseño web, existen multitud de aplicaciones creadas por la comunidad y que están disponibles para descargar. A modo de ejemplo se ha añadido una aplicación llamada "online_status" [12] que permite llevar un control de los usuarios activos en el sistema en cada momento. Esta funcionalidad se ha aplicado únicamente para que usuarios del grupo de Profesores puedan llevar el control.

4.7 Conclusión del desarrollo

Una vez analizadas cada una de las herramientas a utilizar en cada subsistema del proyecto, el sistema completo propuesto para su integración estará formado por las siguientes aplicaciones:

- **cURL** para el subsistema de adquisición.
- **FFmpeg** para el subsistema de codificación y **FLV** como formato a codificar.
- **Red5 Media Server** como servidor dedicado y **RTMP** como protocolo de envío del vídeo.
- **JW Player** como reproductor integrado en la web.
- **Django** como *framework* para el desarrollo de la aplicación web para el control de accesos.

Se ha introducido de manera aproximada en el uso y la funcionalidad de cada herramienta. En los anexos se detalla su instalación, configuración y funcionamiento mucho más en detalle. En el siguiente capítulo se explicará la integración de todas estas herramientas y las pruebas realizadas.

5 Integración, pruebas y resultados

En este capítulo se describe el proceso de integración de las herramientas seleccionadas en el capítulo anterior. Se van a comentar las primeras pruebas realizadas y cómo se han ido solventando los problemas que fueron surgiendo. Por último se presenta la solución final y se evalúan los resultados obtenidos.

5.1 Integración y primeras pruebas

Las primeras pruebas de integración se llevaron a cabo entre las aplicaciones cURL y FFmpeg.

Primero se comprobó que cURL podía descargar correctamente el flujo de vídeo e introducirlo en la entrada estándar de FFmpeg. A continuación se comprobó que FFmpeg podía transformar el flujo entrante *mjpeg* en un archivo *flv*.

Visto que funcionaba, se pasó a probar el *streaming* con un servidor. Se hicieron algunas pruebas con el servidor FFserver (explicado en el punto 4.4.1). Para ello se configuró el archivo "ffserver.conf", ubicado en el directorio /etc/:

```
Port 8090
BindAddress 0.0.0.0
MaxHTTPConnections 2000
MaxClients 1000
MaxBandwidth 1000

<Feed feed1.ffm>
     ACL allow 127.0.0.1
</Feed>

<Stream test1.flv>
     Feed feed1.ffm
     Format flv
     VideoCodec flv
     VideoBitRate 128
     VideoBufferSize 500
     VideoFrameRate 25
     VideoSize 320x240
     NoAudio
     Preroll 5
</Stream>
```

Sabiendo que el *streaming* funcionaba perfectamente, se pasó a sustituir el servidor de pruebas FFserver por el servidor RTMP. Para ello primero se utilizó la aplicación MidiDemo para Red5. Esta aplicación cumplía perfectamente con el cometido de administrar distintos flujos RTMP para luego enviarlos al reproductor. Sin embargo, con la actualización a Ubuntu 14.04 fue imposible volver a hacerla funcionar. Se encontró otra aplicación llamada OflaDemo que se vio que también cumplía los requerimientos para ser incluida en el sistema. Se consiguió instalar y se hicieron pruebas con el reproductor JW Player.

La reproducción del vídeo no era del todo fluida, por lo que se hicieron una serie de retoques en los parámetros de codificación y en el *script* del reproductor hasta que la calidad y la latencia del vídeo fueron las óptimas.

5.2 Integración del reproductor y la web

En las plantillas del Aula, tanto para el grupo de Alumnos como para el de Conferencia, se ha integrado el reproductor JW Player. Para ello se coloca entre los delimitadores de columna central el *script* que incrusta el reproductor.

```
<script type='text/javascript'>
    jwplayer('playerVZcSuaEnViqO').setup({
        file: 'rtmp://localhost/oflaDemo/live',
        width: '100%',
        aspectratio: '4:3',
        primary: 'flash',
        controls: 'false',
        autostart: 'true',
        rtmp: {
            bufferlength: 0.1
        },
    });
</script>
```

A continuación se puede entrar en la web y una vez *logueado* como Alumno con los permisos para entrar en el Aula en la fecha actual se puede observar que el reproductor se ha integrado correctamente:

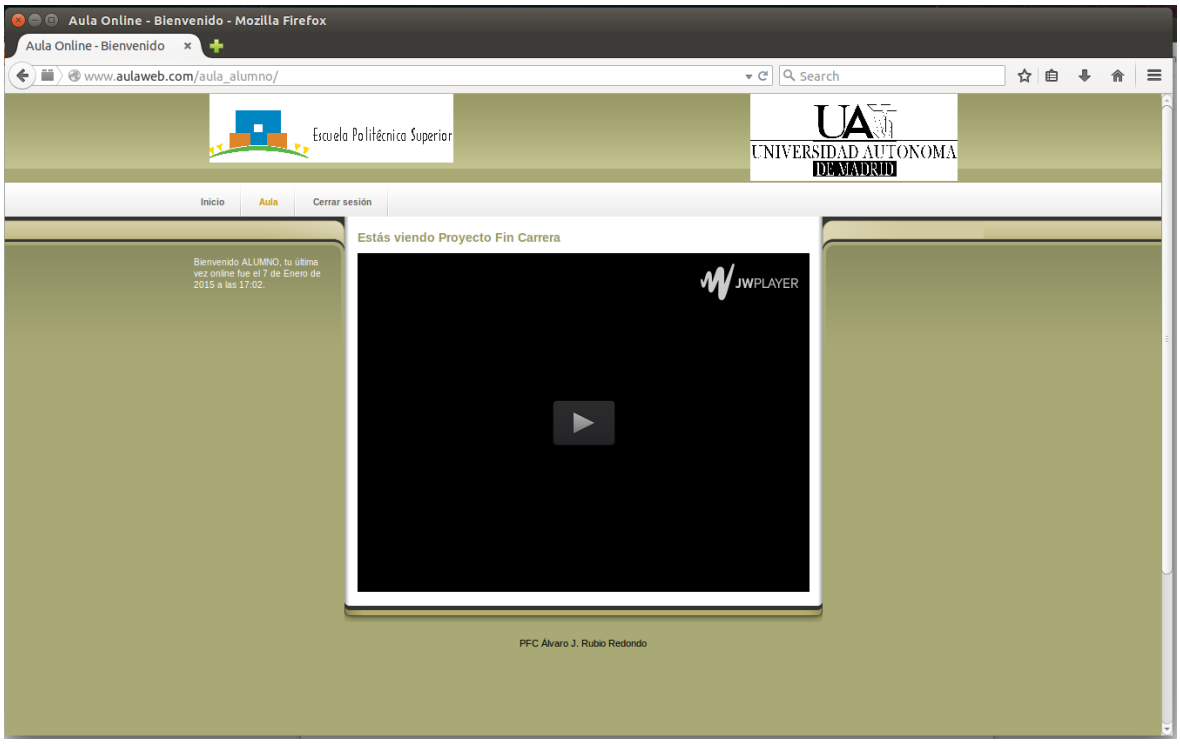

Figura 5-1: Integración del Reproductor en la Web

5.3 Integración del audio en el sistema

Sabiendo que el sistema se comporta como se había previsto, el momento de introducir el audio llegó a ser complicado.

Como se dijo en el punto 4.2.1, cURL no tiene soporte para descargas paralelas. Por lo tanto no se podía usar una sola instrucción cURL para descargar tanto el vídeo como el audio e introducirlos en FFmpeg para posteriormente juntarlos. Finalmente se decidió seguir la estructura que se describe a continuación:

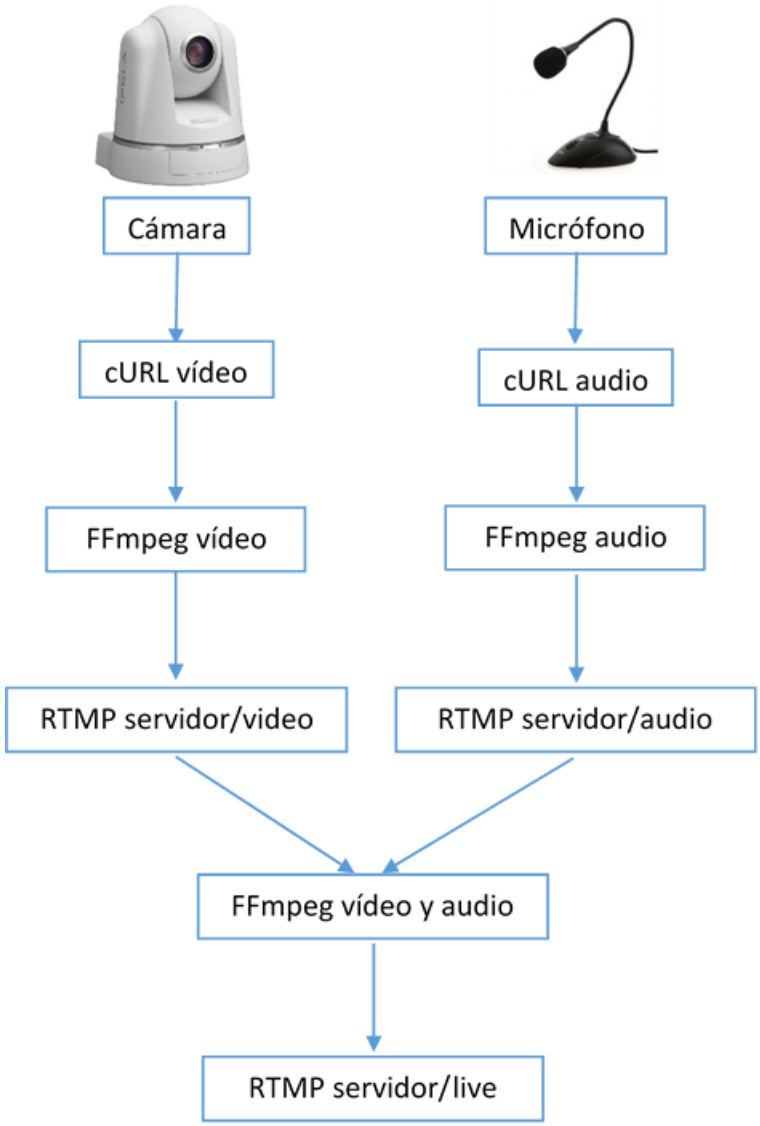

Figura 5-2: Arquitectura adquisición y emisión

El vídeo y el audio se descargan con cURL en instrucciones separadas utilizando la ejecución en segundo plano que proporciona Linux. Se procesan con FFmpeg y se mandan al servidor dedicado por RTMP. Mediante otra instrucción FFmpeg se descargan ambos flujos (ya que FFmpeg sí puede descargar en paralelo), se transforma para obtener la calidad deseada, se juntan en un solo archivo FLV y se vuelve a mandar al servidor por RTMP. Este es el archivo que se reproducirá en la web.

La ventaja de este sistema es que se pueden crear varias calidades de vídeo sin tener que estar conectándose con la cámara para cada una. Adquiriendo el vídeo y el audio y enviándolo tal como se obtienen del servidor de la cámara al servidor que se ha creado da la libertad de poder realizar tantos flujos distintos y con diferentes configuraciones como se desee.

5.4 Integración del sistema completo

Para el sistema final se han creado dos flujos distintos, uno con *bitrate* de 282 kbps (para usuarios del tipo Alumno) y otro flujo con un *bitrate* de 548 kbps (para usuarios del tipo Conferencia). También se ha reducido el tamaño del vídeo ya que la resolución que se obtenía de la cámara era demasiado grande como para ser emitida por Internet.

Por lo tanto, el *script* final para el sistema completo contiene:

- Dos instrucciones de cURL para obtener el flujo de vídeo y audio desde el servidor de la cámara.

- Dos instrucciones FFmpeg que transforman el vídeo y el audio obtenidos por cURL y los envía al servidor Red5 por RTMP.

- Una instrucción FFmpeg que descarga ambos flujos desde el servidor Red5, los une, ajusta la calidad y envía un único flujo FLV con audio y vídeo integrados al servidor Red5 por RTMP, listo para que los clientes lo visualicen.

- Se añade una última instrucción FFmpeg con la misma función que la anterior pero con mayor *bitrate*.

5.5 Despliegue en un servidor real

Hasta el momento todas las pruebas realizadas habían sido con un *host* virtual sobre el mismo ordenador. Este ordenador hacía de servidor y de cliente al mismo tiempo.

Para probar si realmente el sistema funciona se ha implementado en una máquina virtual en los servidores suministrados por el grupo VPULab. Para ello se ha creado una máquina virtual con Ubuntu 14.04 y se han instalado y configurado todas las herramientas tal y como está descrito en los anexos. La dirección final para conectarse al sistema desde cualquier ordenador conectado a Internet es:

http://dymas.ii.uam.es:8081

5.6 Evaluación de los resultados

Las pruebas se han realizado sobre el servidor real y emitiendo dos flujos de vídeo de dos calidades. Se han realizado pruebas conectando varios ordenadores simultáneamente al sistema (10 en total) y no hubo ningún problema. Se observa que, después de 12 horas de utilización ininterrumpida del sistema, el uso medio de la CPU del servidor se mantiene en torno al 50 % y con 613 MB de memoria RAM utilizados (30 %). La tasa de descarga es de 1.4 MB/s, la de subida es de 30 KB/s cuando no hay usuarios conectados, aumentando unos 40 KB/s por cada usuario del tipo Alumno conectado al vídeo y unos 80 KB/s para usuarios del tipo Conferencia. Por lo tanto, el límite de usuarios permitido dependerá del máximo ancho de banda de subida que se tenga contratado. Por ejemplo, si se tienen contratados 10 MB/s de subida, el número máximo de usuarios tipo Alumno reproduciendo el vídeo sería de unos 200, conservando algo de capacidad en caso de producirse picos. Este límite se le especifica en la configuración del servidor web para que no se produzca sobrecarga de usuarios, y se ha dejado el límite en 100 conexiones simultáneas para evitar problemas.

La velocidad de carga de la página y del vídeo es prácticamente inmediata, y en las pruebas realizadas no hubo ningún problema de latencia en la reproducción del vídeo, siendo el movimiento de los objetos de la imagen totalmente fluido y natural. Se observa

que la calidad de la imagen es buena cuando la cámara está estática, pero esta calidad desciende si la cámara se mueve. Dependiendo del uso que se vaya a hacer del sistema, se recomienda ajustar los parámetros de codificación para que prime la calidad frente al ancho de banda o viceversa.

6 Conclusiones y trabajo futuro

En este capítulo se da una visión de qué se ha conseguido en este proyecto y cómo se podría mejorar el sistema en un futuro.

6.1 Conclusiones

En este proyecto se ha desarrollado un sistema para la emisión de clases presenciales en directo a través de Internet. Se ha aprovechado la infraestructura disponible y se han buscado las herramientas que mejor se adaptaban a esta infraestructura y a las especificaciones del proyecto.

Se ha creado una web para que los usuarios que deseen acceder a la retransmisión de la clase no tengan que instalar aplicaciones adicionales un sus ordenadores. Contando con un navegador comercial es suficiente para poder ver el vídeo.

También se ha creado una aplicación que se integra en la web para controlar qué usuarios pueden acceder a la clase y cuáles no. Se han creado varios grupos de usuarios con diferentes permisos y un algoritmo para restringir su acceso al vídeo a las horas de las asignaturas que cada usuario tiene registradas.

Se han cumplido todos los requerimientos iniciales y se ha probado que el sistema funciona en un ambiente real.

Un objetivo muy importante al inicio del proyecto era que el sistema tuviese un mantenimiento sencillo, y gracias a la intensa búsqueda llevada a cabo y a los anexos que acompañan a este documento se puede concluir que este objetivo también se ha cumplido.

Como crítica negativa se podría argumentar que la funcionalidad del sistema es bastante limitada y que no se podría usar en el ambiente académico. En parte es cierto, ya que el sistema desarrollado tan solo es la base. Sin embargo, las herramientas escogidas para llevar a cabo el sistema tienen multitud de configuraciones y de ampliaciones posibles. En el siguiente punto se darán algunas ideas para futuros desarrollos sobre el sistema creado.

6.2 Trabajo futuro

Para hacer al sistema más completo y de cara a implementarlo realmente en el ámbito académico, a continuación se comentan algunas sugerencias e ideas que se podrían llevar a cabo en futuras actualizaciones del sistema

- **Ampliar la funcionalidad de la web:** Se pueden crear más grupos de usuarios con diferentes permisos de acceso o con distintas herramientas. Gracias a que Django es un *framework* muy extendido, existen multitud de aplicaciones que añadirían funciones a la web, como por ejemplo, calendarios, un sistema de mensajes, foros, chats… Un gran repositorio de aplicaciones gratuitas para Django es la web www.djangopackages.com.

- **Añadir nuevas aulas al sistema:** El sistema permite la adición de nuevas cámaras y por tanto nuevas aulas. De esta manera se tendría una especie de campus virtual capaz de estar emitiendo varias clases al mismo tiempo.

- **Sincronización con la plataforma Moodle:** El sistema Moodle ya se encuentra completamente implementado en esta universidad, por lo que se podría aprovechar la gran aceptación que tiene para impulsar el sistema creado en este proyecto. Además Moodle ya tiene una gran base de datos creada con todos los alumnos matriculados en la universidad. Si se pudiese tener acceso a esa base de datos, las tareas de mantenimiento de este sistema serían mucho más sencillas.

- **Mejoras de seguridad:** Dado que el tema de la seguridad no era primordial a la hora de desarrollar el sistema, no se ha profundizado en su estudio. Si se quiere usar el sistema creado en un ámbito real habría que estudiar los protocolos usados y si no fuesen seguros contra ataques externos habría que buscar otros protocolos para sustituirlos.

- **Integración con aplicación de seguimiento:** El sistema tal como está no aprovecha la capacidad de movimiento de la cámara. Una aplicación que

permita automatizar la producción del vídeo sería de gran utilidad. Ya existe una aplicación desarrollada en un TFM en esta universidad [2], pero es necesario mejorar los algoritmos que utiliza, ya que el movimiento de la cámara no es todo lo fluido que se esperaría.

- **Uso en los dispositivos móviles:** Sin duda, si se pudiese acceder al sistema desde dispositivos móviles, sería un gran avance para el sistema. Para ello, se sugiere estudiar a fondo el protocolo HTTP Live Streaming (visto en el punto 4.4.2) e intentar la convivencia con el protocolo RTMP o incluso sustituirlo.

Referencias

[1] **J. L. Gamo Revilla**, «Emisión de clases presenciales: producción automática y aplicación básica,» PFC, UAM, 2012.

[2] **A. González Huete**, «Seguimiento y producción automática mediante cámaras PTZ en entornos de red,» TFM, UAM, 2013.

[3] **W. Taymans, S. Baker, A. Wingo, R. S. Bultje y S. Kost**, «GStreamer Application **Development** (1.4.5)».

[4] **FFmpeg**, «Documentación FFmpeg,» [En línea]. Available: https://www.ffmpeg.org/documentation.html.

[5] **The Apache Software Foundation**, «Apache HTTP Server Documentation Versión 2.4».

[6] **Apple Inc.**, «HTTP Live Streaming Overview,» 2014.

[7] **S. Gong, P. Gregoire y D. Rossi**, «Red5 - Reference Documentation Version 1.0».

[8] **H. Parmar y M. Thornburgh**, «RTMP specification Version 1.0,» 2012.

[9] **M. Lutz**, **Learning** Python, O'Reilly Media, 2009.

[10] **A. Holovaty**, La guía definitiva de Django, Anaya Multimedia, 2009.

[11] **S. Infante Montero**, «Django, el web framework para perfeccionistas,» 2012. [En línea]. Available: http://www.maestrosdelweb.com/curso-django-introduccion/.

[12] **J. Zalewski**, «Aplicación Django online-status,» 2014. [En línea]. Available: https://bitbucket.org/zalew/django-online-status.

[13] **F. Sonnati**, «FFmpeg – the swiss army knife of Internet Streaming,» 2011. [En línea]. Available: http://sonnati.wordpress.com/2011/07/11/ffmpeg-the-swiss-army-knife-of-internet-streaming-part-i/.

[14] **C. Murphy**, HTML y CSS, Anaya Multimedia, 2009.

Glosario

Bitrate	Tasa de bits. Define el número de bits que se transmiten por unidad de tiempo.
Códec	Es una especificación que utiliza un dispositivo o programa para desempeñar transformaciones.
Código abierto	Software distribuido y desarrollado libremente.
Dirección IP	Etiqueta numérica que identifica un dispositivo en una red que utilice el protocolo IP.
FPS	Tasa de cuadros por segundo. (Del inglés Frames Per Second)
Frame	Cada cuadro de la secuencia de vídeo.
Framework	Arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo
IEC	Comisión Electrotécnica Internacional. (Del inglés International Electrotechnical Commission)
ISO	Organización Internacional de Normalización. (Del inglés International Organization for Standardization)
ITU-T	Unión Internacional de Telecomunicaciones, sector de la Normalización de las Telecomunicaciones. (Del inglés International Telecommunication Union)

Lenguaje interpretado	Lenguaje de programación que está diseñado para ser ejecutado por medio de un intérprete.
MVC	El Modelo Vista Controlador es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.
Pipe	Cadena de procesos conectados de forma tal que la salida de cada elemento de la cadena es la entrada del próximo.
PTZ	Cámara que permite movimientos en los ejes horizontal y vertical además de ajustes en el zoom. (Del inglés Pan Tilt Zoom)
Script	Archivo de órdenes para interactuar con el sistema operativo.
Shell	Intérprete de comandos.
Streaming	Transmisión de contenido en tiempo real.
Tipado dinámico	Una misma variable puede tomar valores de distinto tipo en distintos momentos.

Anexos

A. Instalación y funcionamiento de FFmpeg

Para instalar FFmpeg en Ubuntu 14.04 en primer lugar hay que agregar una PPA a los repositorios:

```
sudo add-apt-repository ppa:mc3man/trusty-media
sudo apt-get update
```

Nota: Si se utiliza un proxy hay que sustituir el primer comando por el siguiente:

```
sudo –E add-apt-repository ppa:mc3man/trusty-media
```

A continuación se puede instalar FFmpeg:

```
sudo apt-get install ffmpeg
```

El funcionamiento de FFmpeg es a través de línea de comandos. La estructura de los comandos es la siguiente:

```
ffmpeg –i archivo_entrada [parámetros] archivo_salida
```

Como se ha visto en el desarrollo del sistema, los archivos de entrada y salida no tienen por qué ser archivos locales, también pueden transmitirse a través de protocolos como HTTP, *pipes*, RTP, RTSP, RAW, UDP o RTMP.

Los parámetros soportados son muchísimos, muchos de ellos optativos ya que FFmpeg puede usar el contexto para obtener información, como por ejemplo la extensión del archivo para conocer su formato.

A continuación se presente una lista de los parámetros más útiles [13]:

Parámetro	Explicación	Ejemplo
-f	Especifica el formato de salida (o entrada si se encuentra antes del –i) del archivo.	ffmpeg –i input.mjpeg –f flv output.flv
-i	Establece el archivo de entrada. Se pueden usar varias veces en una instrucción. Para decir que el archivo de entrada está en la entrada estándar se utiliza –i -.	ffmpeg –i input1.mjpeg –i input2.mp3 output.flv
-an	Le indica que se ignore el audio.	ffmpeg –i video.avi –an output.avi
-vn	Le indica que se ignore el vídeo.	Ffmpeg –i video.avi –vn output.avi
-y	Sobrescribe el archivo de salida automáticamente.	ffmpeg –y –i video.avi output.avi
-n	No sobrescribe. Si el archivo de salida existe termina la ejecución.	ffmpeg –n –i video.avi output.avi
-t N	Escribe el archivo de salida durante N segundos.	ffmpeg –i video.avi –t 10 output.avi
-b:a	Especifica el *bitrate* del audio.	ffmpeg –i video.avi –b:a 32k output.avi
-b:v	Especifica el *bitrate* del vídeo.	ffmpeg –i input.avi –b:v 548k output.flv
-c:a	Especifica el códec del audio. Si se usa copy se deja el códec de la entrada.	ffmpeg –i audio.mp3 –c:a copy output.mp3
-c:v	Especifica el códec del vídeo. Si se usa copy se deja el códec de la entrada.	ffmpeg –i video.avi –c:v libx264 output.avi
-r	Especifica los cuadros por segundo del vídeo.	ffmpeg –i input.avi –r 15 output.flv
-s	Especifica el tamaño del cuadro.	ffmpeg –i input.avi –s 320x240 output.flv
-aspect	Especifica el ratio de aspecto del vídeo.	ffmpeg –i input.avi –aspect 4:3 output.flv
-metadata	Añade metadatos de la forma campo=valor	ffmpeg –i input.avi –metadata title=”Video” output.flv
-v quiet	No muestra datos por la terminal	ffmpeg –v quiet –i input.avi output.flv

Tabla A-1: Parámetros FFmpeg

A continuación se muestran algunos usos de FFmpeg con un ejemplo de su sintaxis:

Uso	Ejemplo
Obtener información	ffmpeg –i input > info.txt
Codificar vídeo a FLV	ffmpeg –i input.avi –r 15 –s 320×240 –an video.flv
Codificar de una secuencia de imágenes	ffmpeg -f image -i image%d.jpg –r 25 video.flv
Decodificar un video en una secuencia de imágenes	ffmpeg -i video.mp4 –r 25 image%d.jpg
Extraer una imagen de un vídeo	ffmpeg -i video.avi –frames:v 1 -ss 00:01:00 -f image image-%d.jpg
Extraer el audio de un vídeo	fmpeg -i video.flv -vn -c:a copy audio.mp3
Mezclar audio y vídeo	ffmpeg -i audio.mp4 -i video.mp4 output.mp4
Cambiar el formato contenedor	ffmpeg –i input.mp4 –c:a copy –c:v copy output.flv
Crear un vídeo desde un solo cuadro	fmpeg -loop_input -frames:v 1 -i frame.jpg -t 10s -r 25 output.mp4

Tabla A-2: Usos de FFmpeg

B. Instalación y configuración de Apache y WSGI

Para instalar el servidor Apache junto con WSGI, necesario para montar la aplicación hecha por Django en el servidor, ejecutar el siguiente comando:

```
sudo apt-get -y install apache2 apache2-mpm-prefork
apache2-utils libexpat1 ssl-cert libapache2-mod-wsgi
```

Una vez instaladas todas las herramientas entrar en un navegador web e introducir la dirección http://127.0.0.1. Si sale una página como la siguiente es que no ha habido ningún problema:

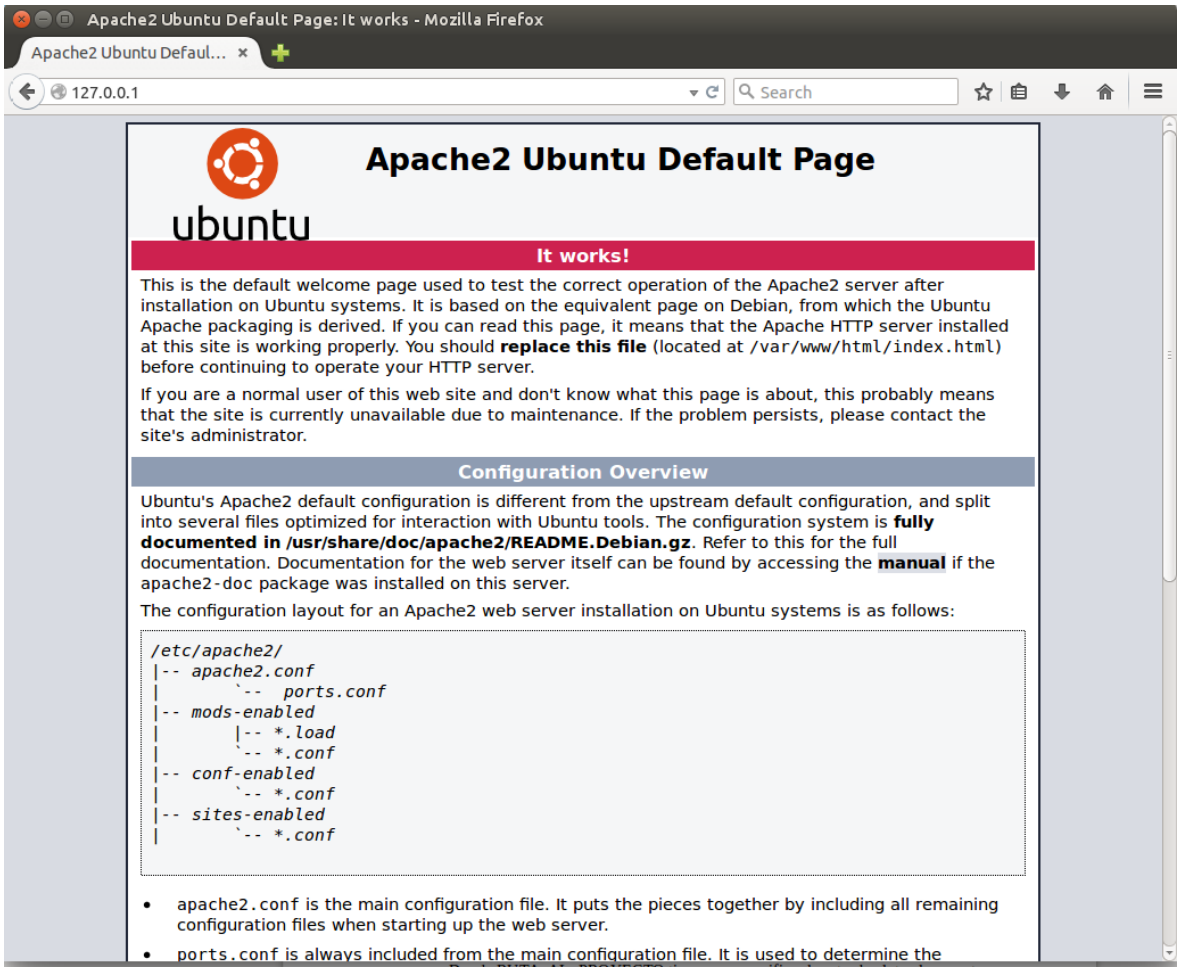

Figura B-1: Apache funcionando

A continuación hay que modificar el archivo /etc/apache2/apache.conf (en versiones anteriores este archivo se llamaba https.conf). En la línea 155 cambiar la palabra *denied* por *granted*:

```
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>
```

Además hay que introducir la siguiente línea para que WSGI funcione correctamente:

```
WSGIPythonPath RUTA_AL_PROYECTO
```

Donde RUTA_AL_PROYECTO tiene que especificar la ruta absoluta al proyecto de Django que se esté desarrollando en estos momentos.

A continuación hay que editar el archivo que configura el *host* que Apache usa por defecto con el siguiente comando:

```
sudo gedit /etc/apache2/sites-available/000-default.conf
```

Cuando se abra el editor copiar lo siguiente, insertando la ruta absoluta al directorio del proyecto cuando sea necesario:

```
<VirtualHost *:80>

    ServerAdmin ADMIN
    WSGIScriptAlias / RUTA_AL_PROYECTO/wsgi.py

    Alias /static/ RUTA_AL_PROYECTO/static/

    <Directory RUTA_AL_PROYECTO>
    <Files wsgi.py>
           Order deny,allow
           #Allow from all
           Require all granted
    </Files>
    </Directory>

    <Directory RUTA_AL_PROYECTO/static/>
           Order deny,allow
           #Allow from all
           Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log

    #Posible values include: debug, info, notice, warn, error,
crit, alert, emerg.

    LogLevel warn

    CustomLog ${APACHE LOG DIR}/access.log combined

</VirtualHost>
```

Reiniciar el servidor Apache y probar el acceso.

De esta manera es posible entrar en la web creada sin usar el servidor de pruebas que Django tiene por defecto.

C. Instalación y funcionamiento de Django

Antes de instalar Django hay que verificar la versión de Python que tiene instalada el sistema. Para ello hay que teclear en una terminal el comando `python`. Las versiones soportadas por Django son la 2.7, 3.2, 3.3 y 3.4.

Si se tiene una versión compatible se puede comenzar con la instalación.

Primero se debe instalar PIP, un gestor de descargas, desde la siguiente web https://bootstrap.pypa.io/get-pip.py. Situarse en el directorio de la descarga y ejecutar el siguiente comando:

```
sudo python get-pip.py
```

Para instalar Django ejecutar el comando siguiente:

```
sudo pip install Django
```

Nota: Si se utiliza un proxy hay que sustituir los dos comandos anteriores por los siguientes:

```
sudo –E python get-pip.py
sudo –E pip install Django
```

Verificar que se ha instalado correctamente ejecutando la terminal interactiva de Python mediante el comando `python` y teclear lo siguiente:

```
>>import django
>>print(django.get_version())
```

Si se han seguido todos los pasos satisfactoriamente debe de mostrar la versión que se ha instalado, que será la última implementada por Django.

En el punto 4.6.5 ya se ha explicado la filosofía de desarrollo que sigue Django. A continuación se van a enumerar algunos comandos útiles para manejar funciones de Django. Éstos se deben ejecutar en el directorio principal del proyecto que se esté desarrollando:

Comando	Descripción
django-admin.py startproject nombre	Crea un proyecto con el nombre especificado.
python manage.py runserver	Ejecuta el servidor para pruebas que tiene Django.
python manage.py shell	Ejecuta el intérprete interactivo.
python manage.py dbshell	Ejecuta el intérprete interactivo de la base de datos especificada en el archivo "settings.py".
python manage.py startapp nombreapp	Se crean los archivos para una aplicación.
python manage.py validate	Comprueba que los modelos de las aplicaciones del proyecto no tengan errores.
python manage.py sqlall nombreapp	Genera el código necesario para crear las tablas de la base de datos
python manage.py syncdb	Crea las tablas en la base de datos. Si se hacen cambios en los modelos, este comando no actualiza la Base.
python manage.py flush	Reinicia la base de datos.

Tabla C-3: Comandos manejo de Django

D. Manual de programador con Django

Los siguientes puntos, además de detallar el proceso de desarrollo de la aplicación web creada, pretenden ser una guía para futuros desarrollos con Django.

D.1 Creación del sistema de ficheros

El desarrollo comienza con la creación del proyecto de la web que se desea crear. Se ejecuta el comando `django-admin.py startproject webPFC`. Esta instrucción crea un directorio con el nombre "webPFC". En el interior de este directorio se creará la siguiente estructura de archivos:

- manage.py
- webPFC
    - __init__.py
    - settings.py
    - urls.py
    - wsgi.py

A continuación se crea la aplicación web que desempeñará la tarea del control de usuarios. Desde la terminal y en la carpeta del proyecto se ejecuta el siguiente comando:

```
python manage.py startapp appWeb
```

Esto creará un directorio y cuatro archivos más, lo que dejaría una estructura de archivos como la siguiente:

- manage.py
- webPFC
    - __init__.py
    - settings.py
    - urls.py
    - wsgi.py

- appWeb
    - __init__.py
    - models.py
    - test.py
    - views.py

D.2 El archivo settings.py

Una parte muy importante del proyecto es el archivo “settings.py”, porque permite configurar la conexión a la base de datos, la zona horaria, el idioma, los directorios principales del proyecto, las aplicaciones del proyecto, etc. Vale la pena dedicarle un tiempo a leer y familiarizarse con el archivo ya que habrá que ir editándolo y cambiando ajustes según se avance con el desarrollo de la web.

D.3 Elección de la base de datos

Django tiene soporte de manera predeterminada para PostgreSQL, MySQL, SQLite3 y Oracle. En este proyecto se va a utilizar SQLite3 por su facilidad de uso.

Al comienzo sólo hay que crear un archivo llamado “app.db” en el directorio del proyecto y modificar en el archivo “settings.py” el apartado DATABASES por lo siguiente:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': 'RutaHastaElProyecto/webPFC/app.db',
        'USER': '',
        'PASSWORD': '',
        'HOST': '',
        'PORT': '',
    }
}
```

D.4 Creación de Modelos

Una vez creado el archivo que contendrá la base de datos hay que editar el archivo "models.py". Este archivo contendrá la declaración de los modelos que luego se convertirán en tablas de la base de datos. Hay que recordar que el lenguaje de programación de las clases será Python y luego será Django el encargado de traducir este lenguaje al propio de la base de datos que se le especifique.

Para esta aplicación basta con crear tres modelos, llamados "Dia", "Asignatura" y "Conferencia".

```
class Dia(models.Model):
    OPCIONES_DIA=(
        ('L', 'Lunes'),
        ('M', 'Martes'),
        ('X', 'Miercoles'),
        ('J', 'Jueves'),
        ('V', 'Viernes'),
        ('S', 'Sabado'),
        ('D', 'Domingo'),
    )
    dia_semana=models.CharField(max_length=1, choices=OPCIONES_DIA)
    hora_ini=models.TimeField()
    hora_fin=models.TimeField()

    def __unicode__(self):
        return u'%s, de %s a %s' % (self.dia_semana,
self.hora_ini.strftime("%H:%M"), self.hora_fin.strftime("%H:%M"))

    class Meta:
        ordering=['dia_semana','hora_ini','hora_fin']
```

El modelo "Dia" sirve para especificar el horario de las asignaturas o eventos que se van a retransmitir. Para crear un nuevo horario hay que especificar el día de la semana, la hora de comienzo y la hora de finalización del evento. La función definida como `__unicode__(self)` se utiliza para que cuando se invoque al modelo, éste devuelva una cadena de caracteres con el formato que se le quiera especificar. En este caso se ha optado por un formato de *Dia*, de *Hora Inicio* a *Hora Fin*.

```
class Asignatura(models.Model):
    nombre=models.CharField(max_length=60)
    siglas=models.CharField(max_length=5)
    website=models.URLField(blank=True)
    curso=models.IntegerField(blank=True, null=True)
    profesores=models.ManyToManyField(User, blank=True,
null=True, related_name="profesores")
    alumnos=models.ManyToManyField(User, blank=True,
null=True, related_name="usuarios")
    dias=models.ManyToManyField(Dia, blank=True)

    def __unicode__(self):
        return self.siglas

    class Meta:
        ordering=['curso']
```

El modelo "Asignatura" contiene los atributos que sirven para definirla, como su nombre, siglas, website, curso y su horario (referenciado al modelo "Dia"). Se definen dos atributos especiales, que son "profesores" y "alumnos". Estos atributos pueden contener varios o ningún User. El modelo User es un modelo que tiene Django en sus librerías internas que provee de funciones muy útiles para manejar los distintos usuarios de la web. Un argumento que se encuentra definido en varios de los atributos del modelo es `blank=True`. Esto indica que el campo será optativo y no será obligatorio que contenga datos.

```
class Conferencia(models.Model):
    nombreConferencia=models.CharField(max_length=60)
    websiteConferencia=models.URLField(blank=True)
    nombreCongreso=models.CharField(blank=True,
max_length=60)
    websiteCongreso=models.URLField(blank=True)
    conferenciantes=models.ManyToManyField(User, blank=True,
null=True, related_name="conferenciantes")
    asistentes=models.ManyToManyField(User, blank=True,
null=True, related_name="asistentes")
    dias=models.ManyToManyField(Dia, blank=True)

    def __unicode__(self):
        return self.nombreConferencia

    class Meta:
        ordering=['nombreConferencia']
```

El modelo “Conferencia” es muy parecido al modelo “Asignatura” salvo que está enfocado a los congresos que se puedan celebrar. Los atributos de este modelo son el nombre de la conferencia, el del congreso, su página web, horario y las personas que van a dar la conferencia y los asistentes a ella.

Para rellenar la base de datos con los modelos creados, se ejecuta en una terminal y desde el directorio del proyecto el comando `python manage.py validate` para verificar que no hay errores de sintaxis. A continuación se ejecuta el siguiente comando para sincronizar la base de datos con los modelos creados:

```
python manage.py syncdb
```

Nota: Si en el futuro se modifican los modelos y se quiere volver a sincronizarlos con la base de datos, el anterior comando no valdrá. Para realizar esta operación primero hay que borrar toda la Base con el comando `python manage.py flush` y a continuación ejecutar el comando de sincronización.

D.5 El portal de Administración

Para insertar nuevas entradas en la base de datos se puede hacer mediante la *Shell* que Django tiene implementada. Este intérprete interactivo permite probar los modelos, hacer consultas, analizar resultados, e insertar nuevos datos. Sin embargo, Django tiene una alternativa mucho más sencilla de utilizar, y es el portal de Administración.

El funcionamiento de este portal se detalla en el Anexo G.3

D.6 Creación de Vistas

Una vista es una función en Python que hace una solicitud web y devuelve una respuesta web. Esta respuesta puede ser el contenido de una página, un error 404, una imagen, un documento XML, entre muchas cosas más.

La vista contiene toda la lógica necesaria para devolver una respuesta. Todas estas respuestas se encuentran en un único archivo y este archivo se llama "views.py", que se encuentra dentro de cada aplicación de Django. A continuación se describen algunas de las vistas implementadas:

`def inicio(request):` Esta función se ejecuta en la página de bienvenida. En ella se da acceso a los usuarios registrados, o se les niega a los usuarios no registrados o que han introducido mal sus credenciales. También gestiona las respuestas a los botones que se implementarán cuando se haga la plantilla, como por ejemplo el botón de registro para los nuevos usuarios.

`def nuevo_usuario(request):` La función se encarga de generar el formulario de inscripción de un nuevo usuario y de guardar sus datos en la base de datos.

`def privado(request):` Cuando un usuario se *loguea* en el sistema, esta función diferencia al usuario según el Grupo al que pertenezca, ya que cada grupo debe tener una página diferente. También realiza una serie de consultas en la base de datos para averiguar las asignaturas que tiene registradas el usuario para mostrárselas en pantalla. Una vez que ha recopilado toda la información se la manda a la plantilla para que ésta gestione cómo mostrarla.

`def aula_alumno(request):` Esta función se ejecuta cuando un usuario *logueado* pulsa en el botón de Aula, para entrar a la sala de visualizado del vídeo. El objetivo de la función es limitar el acceso a la sala a aquellos usuarios que tienen alguna asignatura registrada en el horario en el que se pretende acceder. Para ello se utilizan funciones que Django implementa para conocer el día y la hora actuales. Esta información se compara con las asignaturas que el usuario tiene registradas en la base de datos y si hay alguna coincidencia se le permite el acceso. Similares a esta función son las funciones de "aula_profesor" y "aula_conferencia".

Los permisos de acceso al sistema dependen del grupo al que pertenezca el usuario. Para ello se han creado los Grupos de "Profesores", "Alumnos" y "Congreso" desde el

portal de Administración. Cada uno de estos Grupos tiene unos permisos de visualización diferentes, por lo que habrá que crear plantillas específicas para cada grupo

D.7 Creación de Plantillas

Las plantillas o *templates* son los archivos que permiten la visualización de la página web. Son archivos escritos en HTML, por lo que no se necesita ningún conocimiento de Python para su edición. Estos archivos hay que crearlos dentro de un directorio llamado "templates" en la misma ruta donde se encuentra el archivo "settings.py". Hay que crear una plantilla por cada página que se quiera disponer. Para este proyecto se han creado un total de 16 plantillas mezclando HTML y CSS [14] para tener una apariencia más moderna. A continuación se muestra un ejemplo de las plantillas creadas:

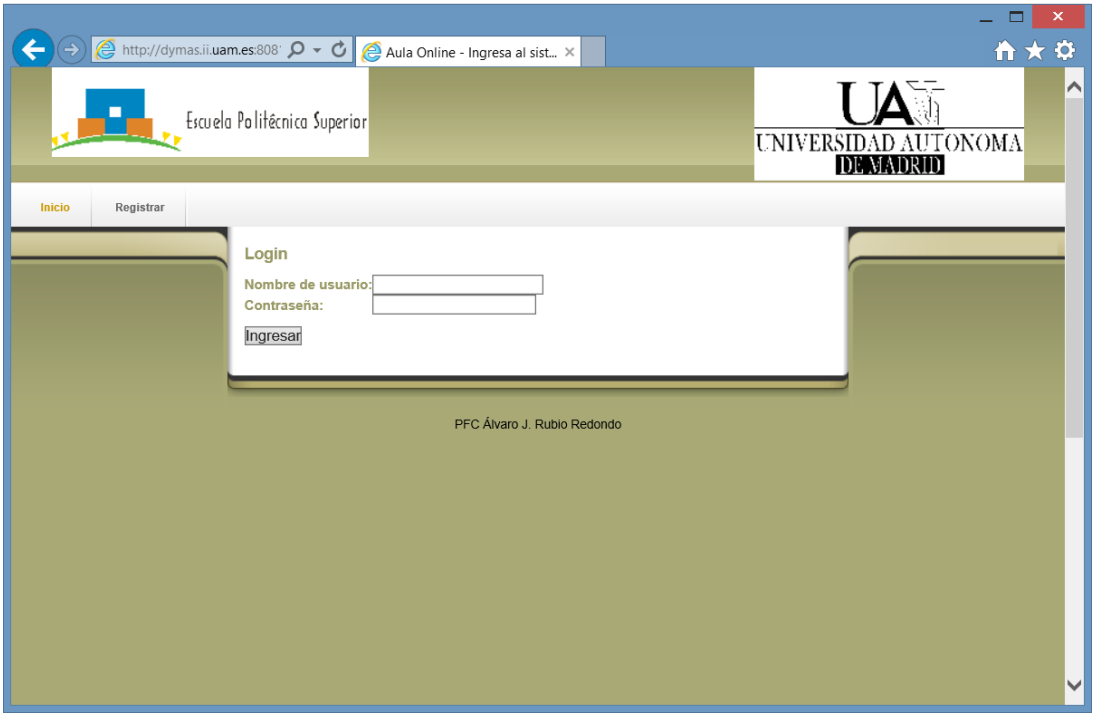

Figura D-2: Plantilla Login

La plantilla anterior es la primera página que se muestra cuando se entra en la web. En ella se da la opción de ingresar en el sistema introduciendo el nombre de usuario y la contraseña o registrarse en el caso de ser un nuevo usuario.

La programación de estas plantillas ha seguido una estructura de bloques para que si en el futuro se quiere editar alguna de ellas para introducir nuevas funciones o crear nuevas plantillas, éstas mantengan la misma apariencia que el resto sin que el programador se tenga que preocupar por la estructura básica de la plantilla. En el archivo "base.html" se encuentra la estructura que van seguir todas las plantillas del proyecto. En este archivo se pueden distinguir los bloques de título, encabezado, columna derecha, columna izquierda y columna central. En la plantilla que se desee crear tan sólo hay que introducir en la primera línea el comando `{% extends 'base.html' %}` y cada bloque delimitarlo entre `{% block` *`nombreBloque`* `%}` y `{% endblock %}`.

Con la finalidad de mantener un orden en el proyecto, se han separado los elementos estáticos como imágenes o archivos CSS de las plantillas. Para ello se crea un directorio en el mismo nivel que el de las plantillas llamado "static". Este directorio se especifica en el archivo "settings.py" para que busque allí los elementos estáticos.

D.8 Despliegue en el servidor web

Cuando ya se ha terminado de editar el proyecto hay que desplegarlo en un servidor. Hay varios métodos para llevar a cabo esta acción, pero en este caso se ha usado la herramienta WSGI en un servidor Apache, ya que es un servidor muy común.

Para ello hay que instalar el módulo para Apache de WSGI y dejarlo configurado (detallado en el B). A continuación hay que editar el archivo "settings.py" indicando la ruta absoluta de la base de datos, la ruta del directorio "static" y cambiar el modo Debug a *false*. Es recomendable cambiar el propietario del directorio principal del proyecto, así como el de la base de datos a "www-data" para que no haya problemas de acceso por parte del servidor.

Con estos pasos la página web ya está lista para su uso.

E. Instalación de Red5

Para empezar hay que instalar unas herramientas de Java mediante el siguiente comando:

```
sudo apt-get -y install openjdk-6-jdk openjdk-6-jre
```

La versión de Red5 de los repositorios de Ubuntu no dio buenos resultados, por lo que se descargó una versión desde los repositorios oficiales de Red5: http://www.red5.org/downloads/red5/1_0_1/red5-1.0.1.tar.gz

Descomprimir mediante el comando `tar -xvf red5-1.0.1.tar.gz`

A continuación mover el directorio a /usr/share/red5:

```
sudo mv red5-server-1.0/ /usr/share/red5
```

Por último es aconsejable crear un *script* para manejar el servidor de manera más cómoda. Para ello copiamos ejecutamos el comando `sudo gedit /etc/init.d/red5` y copiamos el siguiente *script*:

```
#! /bin/sh
# put these contents at: /etc/init.d/red5
# change red5 directory path below as necessary
RED5_DIR=/usr/share/red5
start()
{
echo "Starting Red5 Service"
sudo su root -c "cd $RED5_DIR; ./red5.sh > /var/log/red5.log &"
return
}
stop()
{
echo "Shutting down red5"
sudo su root -c 'killall red5 java'
return
}
case "$1" in
start)
start
;;
stop)
stop
;;
restart)
stop
start
;;
*)
echo "Usage: {start|stop|restart}"
exit 1
;;
esac
exit $?
```

Se le dan permisos al *script* creado mediante `sudo chmod 755 red5` y para iniciar el servidor simplemente hay que introducir en la terminal el comando:

```
sudo service red5 start
```

Para comprobar que el servidor funciona con normalidad, entrar con un navegador en la dirección http://localhost:5080. La página que debe salir es la siguiente:

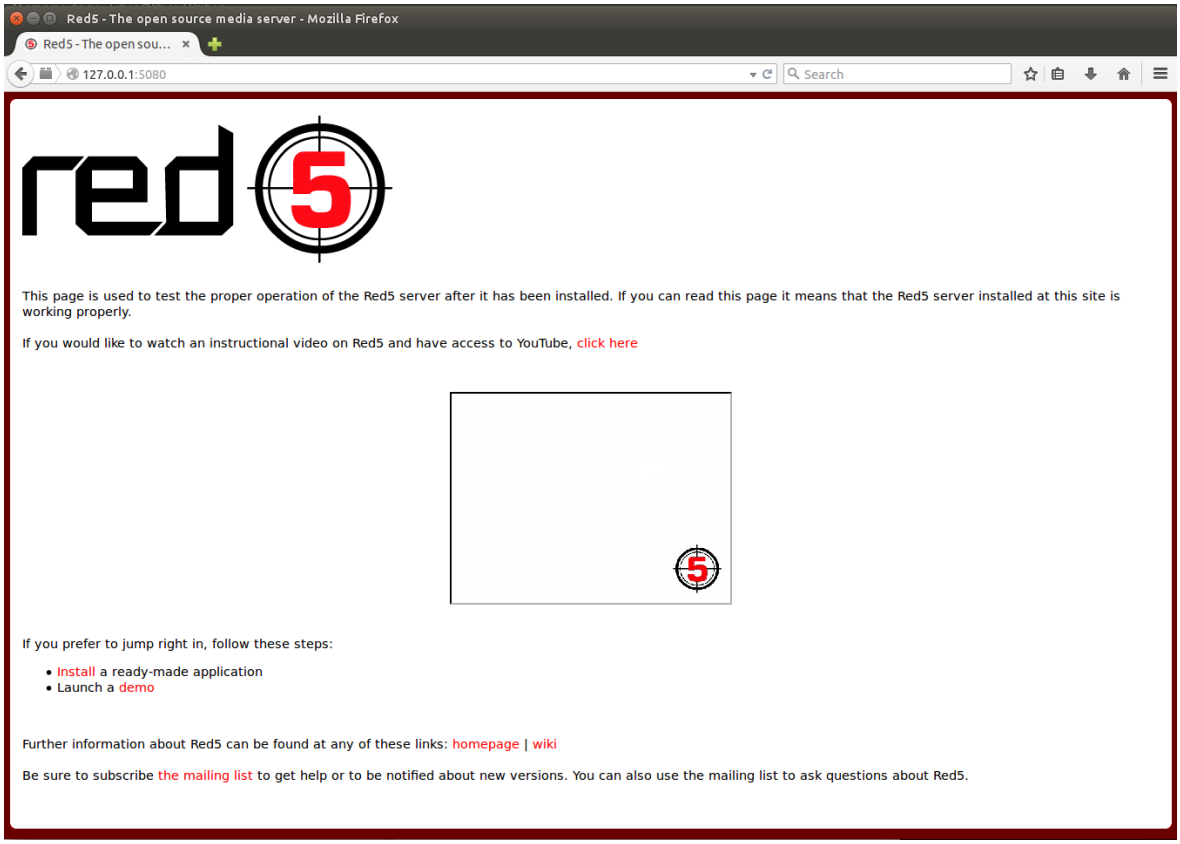

Figura 6.2-3: Portal Red5

Para instalar la aplicación OflaDemo hay que ir a la página del servidor Red5 (http://localhost:5080) y pulsar en la opción Install. Si en la tabla aparece la aplicación OflaDemo pulsar en instalar. Si por el contrario no aparece (depende de la versión de Java instalada) habrá que hacerlo manualmente. Descargar el archivo desde la página http://red5.googlecode.com/svn/snapshots/oflaDemo.war. Extraer el contenido y moverlo al directorio de las aplicaciones de Red5 mediante el comando `sudo mv oflaDemo /usr/share/red5/webapps`.

Por último reiniciar el servidor mediante el siguiente comando:

```
sudo service red5 restart
```

F. Instalación de cURL y SQLite3

La instalación de cURL y la base de datos SQLite3 no tiene ninguna dificultad. Tan solo ejecutar el siguiente comando:

```
sudo apt-get -y install curl sqlite3
```

G. Manual de mantenimiento y administración

Este manual detalla una serie de instrucciones para el mantenimiento del sistema.

G.1 Puesta en marcha del sistema

- Primero se activa el servidor Red5. Desde una terminar ejecutar el siguiente comando:

```
sudo service red5 start
```

- A continuación ejecutar el *script* de captura y envío del vídeo. Para ello ejecutar el siguiente comando en una terminal:

```
./RUTA_AL_DIRECCTORIO/completo.sh
```

- Si se ha hecho algún cambio en la web es recomendable reiniciar el servidor Apache:

```
sudo service apache2 restart
```

G.2 Parada del sistema

- Terminar la ejecución del *script* de envío del vídeo pulsando Crtl+C.
- Parar el servidor Red5 mediante el comando:

```
sudo service red5 stop
```

- Parar el servidor Apache mediante el comando:

```
sudo service apache2 stop
```

G.3 Administración de la web

Para las tareas de administración de la web se ha incluido un portal de administrador que provee Django. Para acceder a él hace falta ingresar la dirección de la página web y agregar /admin.

Saldrá un portal como el de la imagen siguiente:

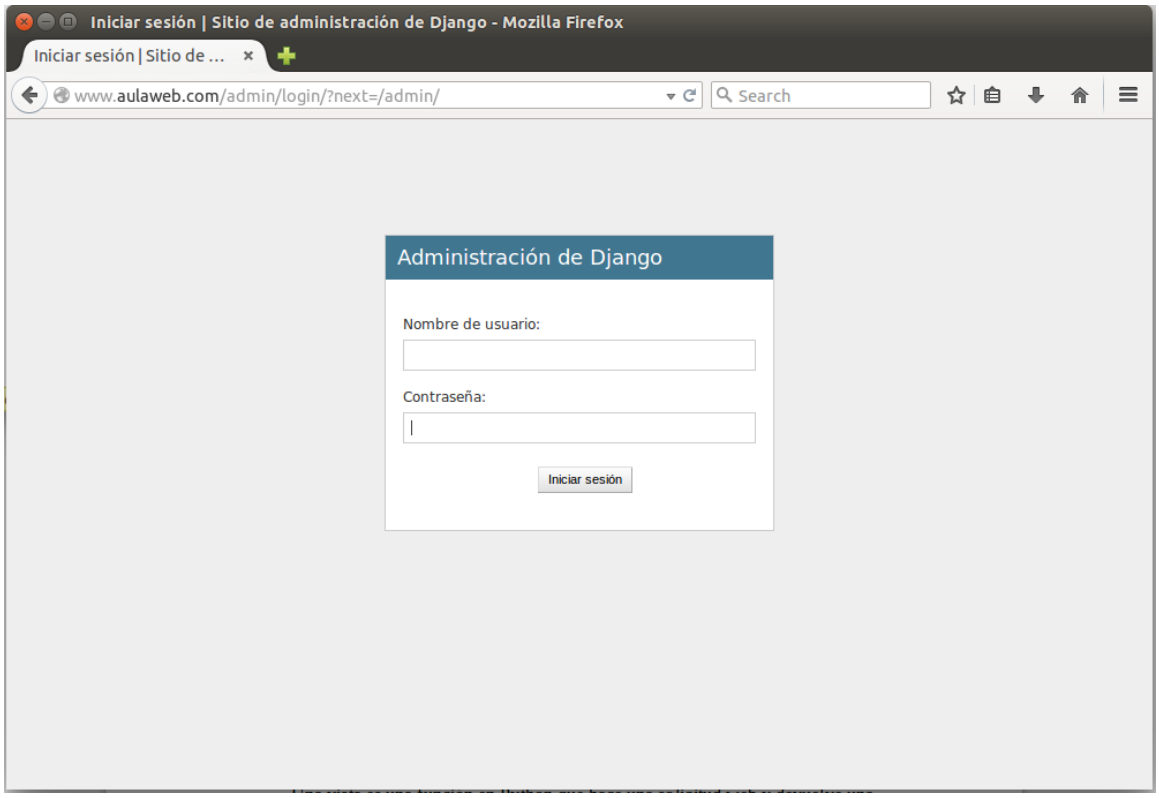

Figura G-4: Portal de Administración Django - Inicio

El usuario con los permisos de administrador se define al crear la base de datos. En este caso se ha creado un usuario llamado “admin” con la contraseña “admin”. Una vez rellenados estos datos se accede al portal:

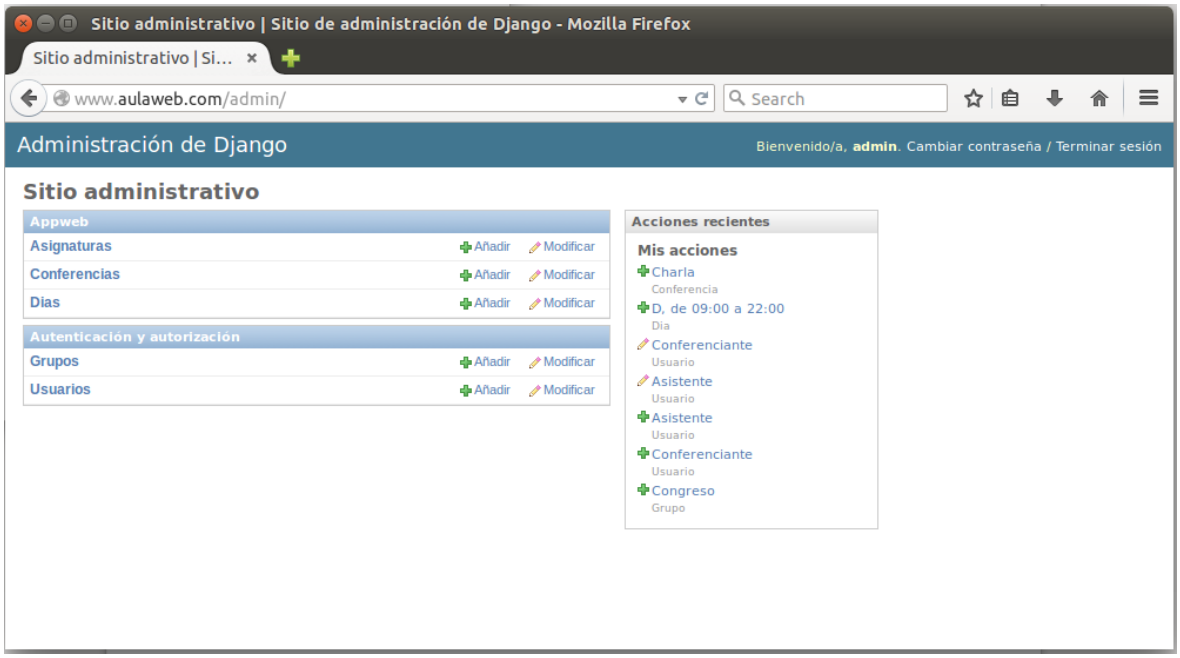

Figura G-5: Portal de Administración Django – Vista general

Desde aquí se puede gestionar toda la base de datos, como crear, modificar o eliminar asignaturas, usuarios y grupos. Este portal hace muy sencilla la tarea de mantener la página web, ya que cualquier persona con los derechos de administración y un nivel medio de manejo de Internet podría hacerlo.

El siguiente sería un ejemplo de cómo agregar una nueva asignatura al sistema:

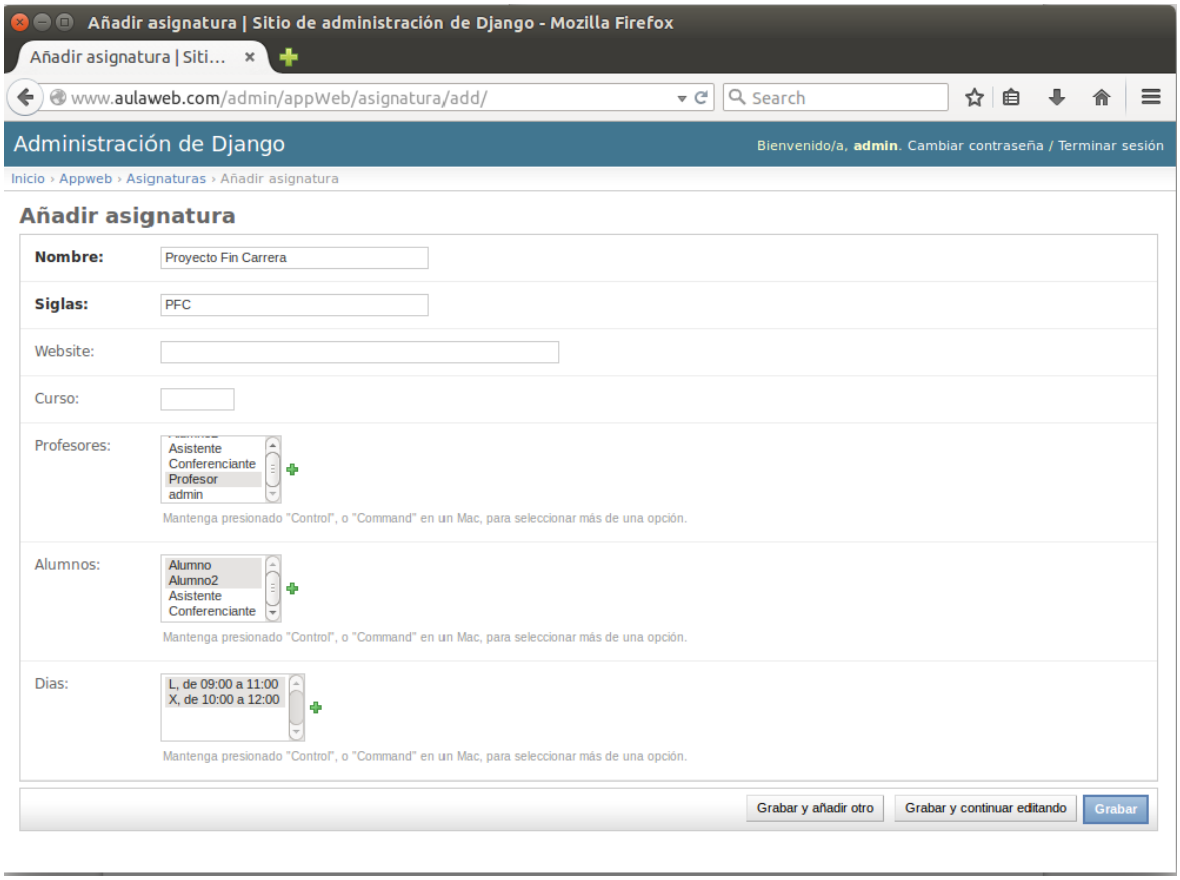

Figura G-6: Portal de Administración de Django – Añadir Asignatura

Los campos en negrita son los obligatorios para la creación de una entrada en la base de datos y los demás son optativos.

Para que los usuarios puedan acceder al sistema correctamente es tarea del administrador darles los permisos correctos. Existen tres grupos creados, que son “Alumnos”, “Profesores” y “Conferencia”. Cuando haya un nuevo usuario, el administrador debe de ir al portal de Administración, entrar en la sección de Usuarios y editar al nuevo usuario. Desde aquí podrá asignarle a un grupo ya creado:

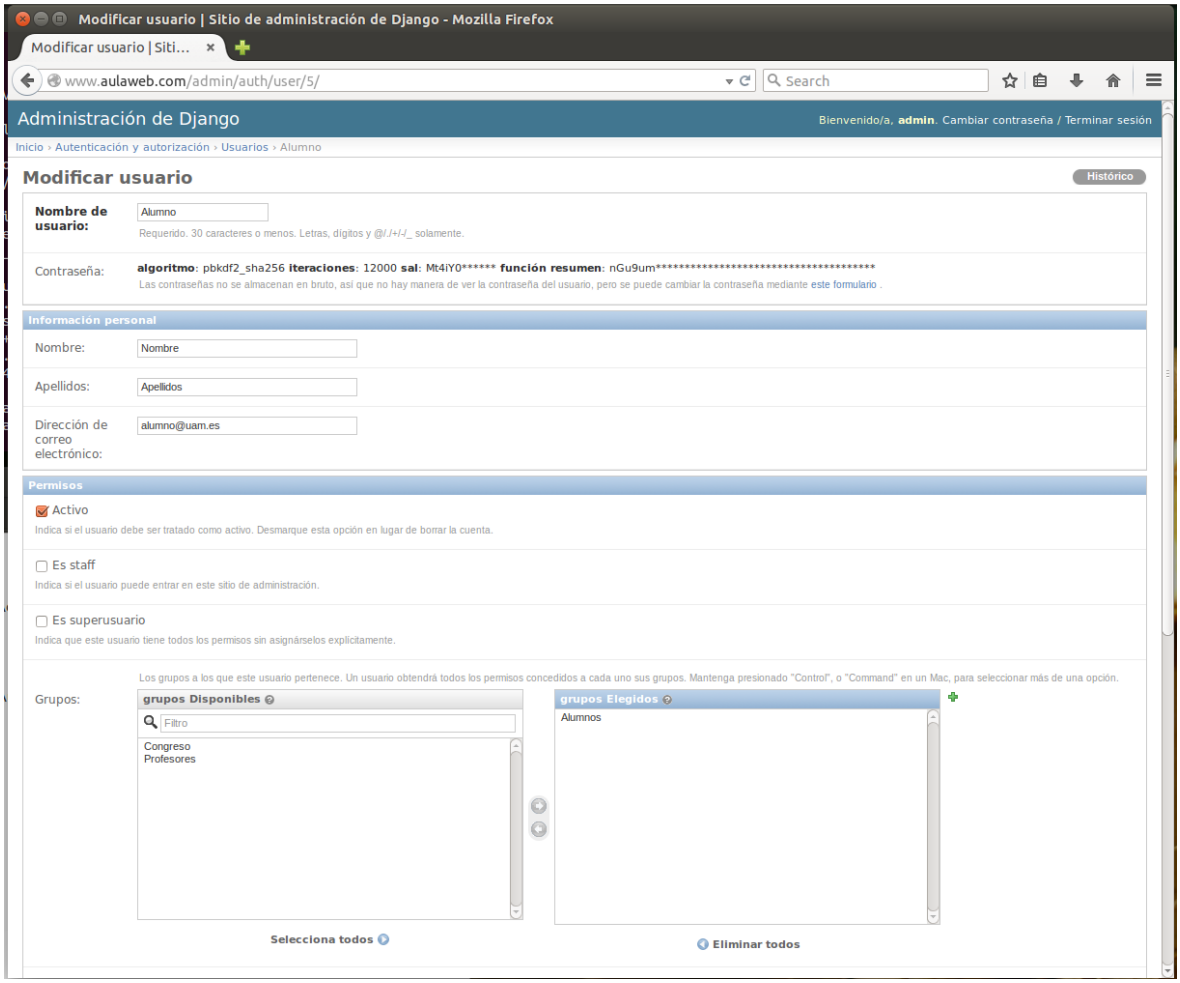

Figura G-7: Portal de Administración de Django – Modificar usuario

PRESUPUESTO

1) Ejecución Material

- Compra de ordenador personal (Software incluido)..... ..... 2.000 €
- Compra de cámara IP..... ..... 2.000 €
- Alquiler de impresora láser durante 6 meses ..... 50 €
- Material de oficina ..... 150 €
- Total de ejecución material ..... 4.200 €

2) Gastos generales

- 16 % sobre Ejecución Material ..... 672 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material ..... 252 €

4) Honorarios Proyecto

- 800 horas a 15 € / hora ..... 12.000 €

5) Material fungible

- Gastos de impresión ..... 80 €
- Encuadernación ..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto ..... 17.404 €

7) I.V.A. aplicable

- 21% Subtotal Presupuesto ..... 3.654,84 €

8) Total presupuesto

- Total Presupuesto ..... 21.058,84 €

Madrid, enero de 2015

El Ingeniero Jefe de Proyecto

Fdo.: Álvaro J. Rubio Redondo
Ingeniero de Telecomunicación

PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un sistema de emisión y gestión de clases presenciales. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al

ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.