Departamento de Telemática

PROYECTO FIN DE CARRERA

Opencast-Matterhorn: Una solución abierta para la captura de clases y su implantación dentro de la infraestructura de la UC3M.

Autor: Christian Cotillas Palomo

Tutor: Isabel Barro Vivero

Leganés, 24 de enero de 2012

Título: Opencast Matterhorn. Una solución abierta para la captura de clases y su implantación en la UC3M
Autor: Christian Cotillas Palomo
Director: Isabel Barro Vivero

EL TRIBUNAL

Presidente: Carmen Guerrero López

Vocal: Jesús Gago Mejía

Secretario: Javier García Guzmán

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día 25 de Enero de 2012 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

A mis padres, por todo el apoyo y ayuda durante toda la carrera.

A Mari Carmen, por la ayuda en la revisión de este proyecto y la paciencia demostrada durante el desarrollo.

A mis amigos, por el apoyo en los momentos más necesarios de la carrera.

A los miembros del Servicio de Informática, por la ayuda ofrecida durante el desarrollo de este proyecto.

Resumen

Este proyecto consiste en la implementación del sistema de grabación y distribución de contenidos Opencast Matterhorn en la infraestructura del Servicio de Informática y Comunicaciones de la Universidad Carlos III de Madrid y su posterior integración en ArcaMM.

De manera general se han encontrado una serie de inconvenientes a la hora de generar y distribuir contenidos audiovisuales generados por la comunidad universitaria, sobre todo en cuanto a calidad técnica, posibilidades de interacción, etc.

Estos inconvenientes se han ido solucionando poco a poco a medida que han ido apareciendo diversas novedades técnicas que permiten visualizar los contenidos generados por la propia comunidad, sin necesidad de estar en la propia universidad.

Actualmente estas grabaciones se suelen realizar mediante complejos sistemas LCS, que automatizan todo el ciclo, realizando la grabación, el procesamiento y la distribución de los contenidos a los canales adecuados.

Palabras clave: distribución de contenidos, streaming, LCS

Índice general

Contenido

Índice de figuras

Índice de tablas

1. Introducción y objetivos

1.1. Introducción

El objetivo de este proyecto es realizar una implantación del sistema Opencast Matterhorn para el Servicio de Informática y Comunicaciones de la UC3M que utilizará el Área de Audiovisuales para la distribución de contenidos.

Este sistema permitirá la distribución de material docente (clases, conferencias seminarios) en multistream (visualizando de manera simultánea tanto al ponente como la presentación) o monostream (visualización de una única señal de vídeo) según la opción elegida.

La implementación requiere por otra parte su integración en el repositorio de contenidos multimedia que dispone la UC3M, la plataforma ArcaMM, donde se aglutina todo el contenido audiovisual generado por la universidad.

1.2. Objetivos

El objetivo fundamental del proyecto es la implantación del sistema LCS (Lecture Capture System) libre y gratuito Opencast Matterhorn, y su integración con el sistema de distribución y publicación de contenidos audiovisuales de la universidad, ArcaMM.

1.3. Fases del desarrollo

El desarrollo del proyecto consta de dos fases, implantación del sistema LCS Opencast Matterhorn, e integración del mismo con la plataforma ArcaMM.

Implementación Matterhorn:

Esta fase consiste en la investigación e implementación del sistema Opencast Matterhorn, incluyendo en la misma todos los aspectos relacionados con la configuración del sistema para su correcto despliegue dentro de la infraestructura de la UC3M.

Integración con ArcaMM:

En esta fase se desarrolla la pasarela entre los vídeos grabados con Matterhorn y la plataforma ArcaMM, diseñando tanto la extracción de metadatos como la interfaz, que permitan categorizar correctamente los vídeos generados en la estructura de ArcaMM.

1.4. Recursos empleados

Recursos Humanos:

- Autor: Christian Cotillas Palomo
- Director del proyecto: Francisco Cruz Argudo
- Tutora del proyecto: Isabel Barro Vivero

Recursos Materiales:

- Máquina Virtual con procesador Intel Xeon E5420a 2.50GHz, 2Gb de memoria RAM, 10Gb capacidad de almacenamiento.
- Almacenamiento 500 GB vía NFS
- PC con procesador Intel Core i7 860 a 2.80GHz, 2Gb de Memoria RAM y 900Gb almacenamiento en Disco
- Capturadora VGA Epiphan VGA2USB LR
- Capturadora Vídeo Hauppauge WinTV PVR-350
- Conexiones con las salidas de audio y vídeo de la sala de audiovisuales

Recursos Software:

- Sistema operativo Ubuntu 9.10 para máquinas virtuales
- Sistema operativo Ubuntu 10.04 para PC Capturador
- Servidor Apache 2
- Gestor de Base de Datos MySQL
- PHP 5
- Editores de programación CodeLite (Linux) y Notepad++ (Windows)

1.5. Estructura de la memoria

La presente memoria está compuesta de 8 capítulos (capítulo de introducción incluido) para facilitar su lectura y comprensión. A continuación se incluye un breve resumen de cada capítulo.

Capítulo 1: Introducción y objetivos

Este capítulo muestra una breve descripción del proyecto a realizar, así como otros datos de interés general.

Capítulo 2: Estado del Arte

En este capítulo se expone la situación actual, tanto de los sistemas de grabación de contenidos audiovisuales, como de la situación particular de la UC3M.

Capítulo 3: Opencast Matterhorn

En este apartado se presenta una visión completa del sistema LCS Opencast Matterhorn, su arquitectura, así como su configuración para el correcto funcionamiento del sistema.

Capítulo 4: Pasarela ArcaMM

En este capítulo se describen los aspectos relacionados con la pasarela Matterhorn-ArcaMM, incluyendo el extractor de metadatos, la interfaz web y el WebService de comunicación.

Capítulo 5: Infraestructura utilizada

En este capítulo se describen ampliamente todos los elementos hardware y software que se han utilizado en el proyecto.

Capítulo 6: Desarrollo del proyecto

En este capítulo se lleva a cabo un desarrollo pormenorizado de todo el proceso dedicado a la realización del proyecto, desde la primera instalación de Opencast hasta la integración del sistema de la pasarela en los servidores de Audiovisuales, pasando por todas las fases intermedias. También se detallan las soluciones adoptadas ante las dificultades que aparecían en su elaboración.

Capítulo 7: Conclusiones

En este capítulo se exponen una serie de conclusiones finales relativas a este proyecto: debilidades encontradas, oportunidades y fortalezas, líneas futuras, etc.

Capítulo 8: Presupuesto

En este capítulo se describen los costes relacionados con el proyecto.

2. Estado del Arte

2.1. Evolución Histórica

La historia de la enseñanza ha pasado por muchas fases, diferentes metodologías y didácticas que han ido evolucionando a lo largo del tiempo.

Un hito clásico son las clases magistrales, vigentes aún en la actualidad en algunas enseñanzas. En estas, el docente transmite los contenidos a los alumnos, utilizando de manera general como soporte una pizarra o rotafolios.

Los avances tecnológicos han permitido modificar este paradigma, permitiendo, por ejemplo, mostrar a los estudiantes fotografías, diagramas y otros recursos, utilizando para ello los retroproyectores que permitían mostrar en una pared o pantalla las transparencias realizadas sobre hojas transparentes de acetato.

Más adelante, la aparición y estandarización de los ordenadores personales permitió mejorar dichas transparencias, tanto en su aspecto, como en la incorporación de mayor interactividad en las exposiciones

Las aulas, poco a poco fueron aumentando su equipamiento, incorporando pantallas, micrófonos, altavoces, en resumen, un completo sistema de audio y vídeo que permitió que las explicaciones adquirieran una nueva dimensión. No era simplemente

pasar de una diapositiva a otra, las explicaciones podían requerir y permitían el uso de material audiovisual con el que ilustrar el tema que el ponente estuviera comentando.

Más tarde se incorporó el siguiente paso: la posibilidad de grabar las clases. En principio se realizaban las grabaciones de una forma primigenia, (grabando al profesor/ponente) y pudiendo visualizar al fondo la pizarra o pantalla en la que estuviera explicando la ponencia que se estuviera impartiendo.

Este proceso aunque de forma rudimentaria, permitió que por primera vez se pudiera atender a la clase, seminario o conferencia, sin necesidad de estar físicamente en el aula donde se impartiera, simplemente visualizando a posteriori la grabación. Lógicamente estas grabaciones tenían ciertas limitaciones, pero era un comienzo. Con el tiempo, estos sistemas mejoraron, permitiendo también grabar con cierta calidad la ponencia que se estuviera impartiendo.

La evolución cronológica generó aparte de la evolución tecnológica (nuevos códec de vídeo y audio, ordenadores personales más potentes y un aumento en el ancho de banda de las redes de datos) la aparición de nuevas formas de comunicación, como las videoconferencias, que permiten que los estudiantes de un determinado curso se encuentren a kilómetros de distancia del aula física donde se encuentra el profesor, o incluso que un mismo curso, sea impartido simultáneamente a personas de universidades distintas, interactuando entre ellos, como si, y salvando las distancias, se encontraran en una misma aula. Esto se traduce en que la ubicación tanto de los alumnos como del profesor deja de tener una relevancia significativa.

No obstante, estos sistemas presentan el problema de cómo llevar a cabo la distribución de los vídeos grabados, vídeos que posteriormente hay que subir a alguna página para su publicación.

Uno de los sistemas más utilizados para distribuir los vídeos grabados es subir estas grabaciones en los sistemas **LMS** (Learning Management System). Se trata de Sistemas de Gestión de Aprendizaje utilizados para la docencia, por ejemplo Moodle o Sakai, en los que las grabaciones sirven de apoyo para el curso, funcionando como fuente de consulta extra, además de los consabidos y tradicionales apuntes.

Bajo estas premisas, aparecieron los sistemas LCS (Lecture Capture System), sistemas que aparte de integrar en una aplicación las herramientas necesarias para la grabación, incorporaban plataformas de distribución de contenidos, con el fin de poder visualizar las grabaciones realizadas.

Algunas de estas plataformas permiten la visualización en **multistream** (es decir visualizar dos fuentes de vídeo simultáneamente, en este caso, la grabación del ponente y de la ponencia). Esta característica permite visualizar desde cualquier PC, una clase como si se encontrara en el lugar físico donde se imparte, pudiendo ver tanto el contenido, como el docente, de manera simultánea.

Algunas de estas alternativas, que integran la grabación y distribución son:

- Panopto
- SWITCHcast
- ePresenter
- Podcast Producer 2

Estas plataformas permiten la visualización en el momento que se desee (Vídeo on Demand, VoD). Como valor añadido, se puede destacar que hoy en día, con el auge de los dispositivos móviles (smartphones y tablets principalmente), no es necesario que el oyente se encuentre delante de un ordenador, puesto que se puede acceder al contenido deseado, de entre los recursos que se encuentren disponibles, en el momento y lugar que se quiera, con independencia del dispositivo de acceso elegido.

Como último punto mencionar que si bien en principio, todos estos “artificios”, estas “nuevas tecnologías” (en su correspondiente momento) se encontraban restringidas en aulas especialmente diseñadas, la evolución de la tecnología ha permitido eliminar esta limitación, pudiendo sacar estos sistemas de dichas aulas y trasladándolas a aulas corrientes, o incluso al uso de sistemas portables, que permiten realizar grabaciones de este tipo sin depender de que la misma se realice en un aula determinada.

La mayoría de estas plataformas disponen de un coste elevado y el interés del Área de Audiovisuales, es la de incorporar a sus sistemas la plataforma OpenCast Matterhorn, un sistema LCS de libre distribución englobado dentro de las nuevas tecnologías que se van incorporando a la docencia en la actualidad.

2.2. Situación en la UC3M

En la UC3M el Servicio de Informática y Comunicaciones (SdIC), y más concretamente el Área de Audiovisuales, se encarga de la grabación y difusión de eventos, clases, seminarios, conferencias, etc. En esta Área se ha desarrollado este proyecto fin de carrera.

Para la difusión de los contenidos grabados, el Área de Audiovisuales dispone de la plataforma ArcaMM, portal de difusión de vídeos de desarrollo propio, inspirada en el Agregador RSS de Contenidos Académicos (ARCA) plataforma originalmente desarrollada por la UC3M pero que ahora es gestionada por RedIris. Su uso está extendido a otras universidades españolas, las cuales agregan metadatos de las grabaciones que realicen y que quieran poner a disposición de la comunidad universitaria en general.

En cuanto a ArcaMM, es el portal de la Universidad dedicado a la distribución y publicación del material grabado y emisiones en directo. ArcaMM está compuesto por un front-end y un back-end, que se describen a continuación.

El "front-end" del portal permite la de búsqueda por categorías, series, eventos en directo, etc.

El "back-end" consiste en un completo panel de control, que permite seleccionar qué vídeo se quiere publicar o retirar y comprobar las características técnicas del vídeo en cuestión, además permite editar los campos que se estimen necesarios, ya sea serie y/o categoría en la que se quiera englobar el vídeo, autores, idioma, licencia, si dispone de subtítulos etc. También permite la posibilidad de exportar y publicar en plataformas educativas externas como: Arca, iTunesU, youtube.edu y sistemas recolectores basados en OAI-PMH.

Una vez realizada la grabación, comprobada su calidad y aceptada su publicación, el vídeo queda disponible a los usuarios del portal para su visión y si se ha estimado conveniente, también para su descarga.

La forma de realizar estas grabaciones, era hasta el curso 2009/2010, mediante la plataforma Windows Media Services. Este sistema grababa únicamente la señal que llegaba por una única entrada de vídeo, por lo que en caso de querer grabar tanto al presentador como a la presentación que se estuviera realizando era necesario disponer de algún sistema externo, como una mesa mezcladora de vídeo, donde se realizaba la conmutación entre señales de vídeo.

Esta forma de grabación, que utilizaba tarjetas PIP, presentaba una serie de limitaciones. Por ejemplo, no existía la navegación por diapositivas, con lo que había que avanzar y retroceder el vídeo en busca de la parte que interesara.

Otra de las limitaciones se debía a la propia grabación en PIP, ya que al alternar los tamaños del vídeo del presentador y de la grabación, se podía perder fácilmente el hilo de la explicación, etc.

Los vídeos grabados con este sistema tienen, principalmente, el formato Windows Media Vídeo (*.wmv).

Tras su grabación, los vídeos son editados con el fin de eliminar partes innecesarias, corregir posibles errores, tomas falsas, etc. Una vez editados los vídeos, quedan listos para su catalogación en el sistema ArcaMM, donde se añade el vídeo incluidas sus propiedades y características técnicas, tras su análisis con la herramienta **MediaInfo** que proporciona estos datos y también se definen los distintos portales de publicación así como la visibilidad de los contenidos

Una vez completados esos datos se puede continuar la catalogación y posterior publicación. La visualización de estos archivos se realiza mediante el reproductor FlowPlayer, que permite la reproducción de diversos tipos de archivos a través de streaming.

En resumen, este sistema de grabación basado en Windows Media Services funciona, pero proporciona una experiencia de usuario muy pobre y poco acorde con lo que otros sistemas pueden proporcionar hoy en la actualidad.

En el curso 2011/2012, se sustituye el programa de grabación por Adobe Flash Media Encoder. El proceso de grabación se sigue realizando de la misma forma, excepto que los archivos generados son archivos de vídeo codificados con el códec VP6 o H.264 y contenedor Flash flv o f4v.

El formato anteriormente usado, *.wmv, se había quedado ya obsoleto, no proporcionaba la calidad requerida y la reproducción a través del portal no funcionaba correctamente en las plataformas Linux y Apple, ya que los complementos necesarios no funcionaban como en plataformas Windows.

Además, el contendor Flash aporta mejoras en cuanto a la reproducción por streaming, resultando por tanto una alternativa superior a los archivos *.wmv utilizados anteriormente, tanto en prestaciones como en calidad.

Por otro lado, se han implantado otros sistemas útiles para la docencia como el sistema Adobe Connect, que proporciona un sistema de aulas virtuales, con presencia vía webcam de varios participantes, así como la posibilidad de compartir transparencias, chat, etc.

Los actos realizados a través de Adobe Connect también puede ser grabados e incluso incorporados como grabaciones a ArcaMM, permitiendo poder ver a posteriori una reunión, conferencia, etc., realizada a distancia por varios participantes.

El Área de Audiovisuales se ha mostrado interesada en sistemas que permitan poner a disposición de los usuarios los vídeos que genera la comunidad universitaria, con la menor intervención directa posible y que fueran escalables, por lo que se comenzó a buscar sistemas de gestión de contenidos y distribución, que permitieran la grabación y difusión tanto de audio/vídeo, como de gráficos en sincronía, por lo que se fijaron en los sistemas LCS (Lecture Capture System). Estos sistemas incorporan tanto la grabación como el procesamiento y difusión de dichos vídeos.

Las grabaciones realizadas, posteriormente requerirán de un procesamiento para la incorporación de datos a ArcaMM, con el fin de tener un repositorio centralizado de

todos los vídeos generados por la comunidad universitaria, aunque la visualización se realice en el propio reproductor del LCS.

La primera solución de este tipo que dispone la universidad es el sistema **MediaSite**, de Sonic Foundry, que es otro sistema LCS.

Sus características principales se detallan a continuación:

- Automatización de la captura y catalogación.
- Emisión en directo
- Subtitulado
- Navegación por diapositivas
- Escalabilidad, permite aumentar la infraestructura fácilmente y sin requerir un mayor control
- API de Control

No obstante y a pesar de las ventajas que proporciona MediaSite, el elevado coste de esta herramienta hace que se busquen otras alternativas de prestaciones similares, optando por la iniciativa Opencast y el desarrollo de Matterhorn, un sistema LCS libre, sin coste por licencia.

Este proyecto versará sobre la implantación en la UC3M de la plataforma Opencast Matterhorn, así como su integración en el sistema ArcaMM del Área de Audiovisuales.

3. Opencast Matterhorn

Opencast Matterhorn es un nuevo sistema LCS (Lecture Capture System), diseñado para proporcionar a la comunidad universitaria una plataforma de código libre y sin coste por licencia.

Este sistema, a pesar de ser relativamente nuevo, se ha mostrado desde el inicio como un sistema potente y altamente configurable, adaptable por tanto a las necesidades formativas de uso educativo.

En contrapartida, su configuración no dispone como otras alternativas de una GUI amigable para dicha tarea, que permita una fácil gestión integral del sistema.

Por tanto, una de las tareas más importantes de este proyecto es comprender el funcionamiento de Matterhorn desde el interior para poder usar este sistema con todo su potencial. Esto implica estudiar y comprender desde el funcionamiento del fichero de configuración del primer servidor, hasta la generación de feeds RSS y ATOM para distribución de contenidos, pasando por la configuración de codificaciones, definición de workflows, configuración de vídeo segmentación, etc.

3.1. Panorámica del sistema

Opencast es un proyecto de ámbito internacional desarrollado por instituciones de educación superior que tiene como objetivo la creación y mantenimiento de software para la grabación de clases magistrales y su posterior procesado. Actualmente el

proyecto más destacado nacido de esta comunidad es el proyecto Opencast-Matterhorn.

Opencast Matterhorn es un sistema libre de gestión de contenidos y webcasting que automatiza la captura, gestión segura, suministro y búsqueda de contenidos enriquecidos (audio + vídeo + gráficos en sincronía) para su visualización en vivo o bajo demanda. Este sistema es similar a otras alternativas privativas como MediaSite de Sonic Foundry o Podcast Producer 2 de Apple Inc.

El objetivo es poder disponer de una alternativa a estos sistemas de gestión de contenidos privativos, pero más económica, ya que el coste de las licencias es alto en el caso de MediaSite, y en cuanto a Podcast Producer, obliga a disponer un sistema Apple equipado con Mac OS X Server. Esta última alternativa presenta también limitaciones tanto a la hora de distribuir las grabaciones como de modificar los formatos utilizados, ya que el objetivo principal de esta herramienta es la creación de Podcast y por tanto las opciones de configuración se encuentran limitadas.

Opencast Matterhorn es un sistema altamente escalable, que permite, en caso de que se requiera, desplegar fácilmente más servidores y capturadores para aumentar las capacidades del sistema, manteniendo el control total del despliegue desde un único equipo.

La plataforma fue creada inicialmente por la Universidad de Berkley que generó a su vez la Opencast Comunity (Comunidad Opencast), comunidad que aglutina a todas las entidades de educación interesadas en el proyecto, que pueden participar en su desarrollo.

Recientemente, y debido al agotamiento de los fondos de los que disponía el proyecto Opencast gracias a subvenciones (como la de la Fundación Hewlett), se ha modificado la forma de funcionamiento de la comunidad. Esto ha provocado una reorganización en el gobierno del proyecto [Ver **Anexo 4**]. Para ello, la Comunidad Opencast pone a disposición de los participantes tanto listas de correo como un *tracker* para notificar bugs, problemas encontrados, etc.

En España se ha creado también una comunidad Opencast en español, de la que forman parte entre otras, la Universidad de Vigo, la Universidad Politécnica de Valencia y la Universidad Carlos III de Madrid. Esta comunidad consta de una lista de correo propia para tratar temas relacionados con Matterhorn en temas de soporte al sistema, etc., y permite la organización de videoconferencias entre todos los participantes, para la puesta en común de experiencias o problemas detectados en el sistema.

3.2. Qué permite

Opencast Matterhorn permite la grabación y distribución de contenidos audiovisuales a través de streaming de una forma asequible, ya que únicamente es necesario pagar por el equipamiento utilizado y no por el propio sistema.

Asimismo permite, gracias a su sistema de feeds, incorporar grabaciones realizadas en Opencast Matterhorn a cualquier página, ya sea insertando la grabación directamente, enlazando a la misma, etc.

El sistema permite codificar los vídeos en distintos formatos para facilitar su visualización en cualquier tipo de dispositivo (PC, tableta, Smartphone, etc.).

Debido a la arquitectura con la que fue concebido, Opencast Matterhorn es un sistema fácilmente escalable, que permite desplegar más servidores y capturadores de forma sencilla para aumentar la capacidad de proceso del sistema.

3.3. Requerimientos

Para instalar un sistema Matterhorn es recomendable el uso mínimo de 3 equipos, ya sean virtuales o físicos. También es factible realizar toda la instalación de Matterhorn en un único equipo, pero debido a su menor rendimiento, no resulta muy recomendable, ya que existen operaciones que requieren el uso casi exclusivo de los recursos del ordenador, dejando al resto de operaciones con recursos limitados.

En principio la estructura básica de un sistema Matterhorn es un servidor como Administrador, otro encargado de la distribución de contenidos (servidor streaming, engage) y un tercer servidor encargado del procesamiento de los vídeos (Worker). Es recomendable que este último equipo, que se dedica a tareas de procesamiento de vídeo, disponga de la suficiente capacidad de proceso así como de memoria suficiente para efectuar dichas tareas, ya que las operaciones relacionadas con vídeos requieren gran capacidad de procesamiento. Se puede observar un ejemplo de arquitectura básica de un sistema Matterhorn en la **Ilustración 1**.

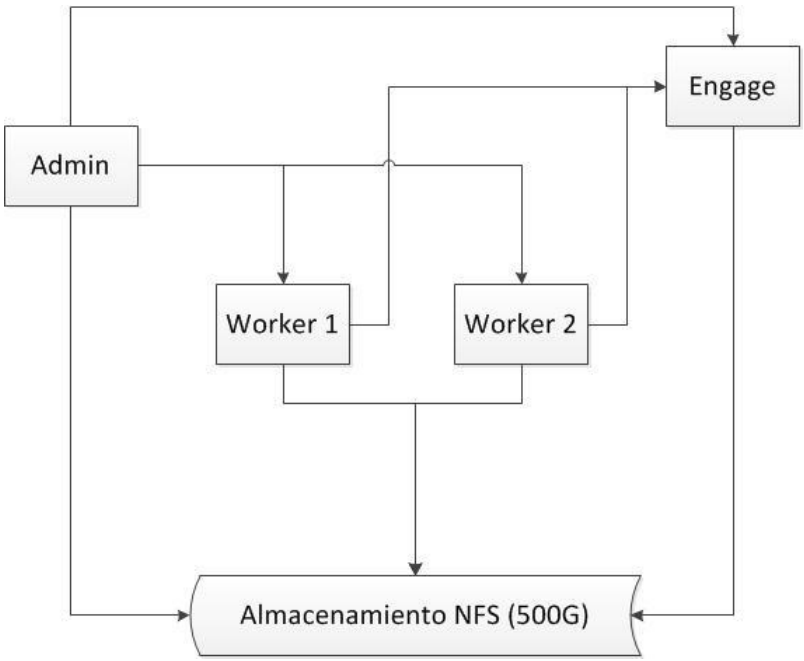


Ilustración 1 Esquema Servidores Matterhorn

A partir de la versión 1.1, se ha mejorado el balanceo de carga de los servidores "*Worker*", permitiendo disponer de varios equipos para esa tarea, lo que reduce el tiempo necesario para el procesamiento de los vídeos.

También es necesario que el sistema disponga de un espacio de almacenamiento compartido, en este caso por NFS, donde se almacenan tanto los ficheros de trabajo que se están procesando, como los ficheros que se generan para su distribución.

Lo más recomendable es que este almacenamiento compartido sea lo más directamente accesible por los servidores de trabajo, ya que de lo contrario el tiempo necesario para el procesamiento de los vídeos se verá penalizado.

3.4. Implantación Servidores

El sistema Opencast es una plataforma desarrollada en Java que emplea la herramienta Apache Maven para su desarrollo y funciona sobre el sistema Apache Félix (Ver **Software Servidores**.), el cual proporciona el funcionamiento de Matterhorn sobre OSGI (Ver **Glosario**). Estas, por tanto, son herramientas que se han necesitado utilizar en el presente proyecto.

Para su implantación en la universidad, se cuenta con la siguiente infraestructura:

- 4 máquinas virtuales con procesador Intel Xeon a 2.5 Ghz, dos de ellas con 1Gb de memoria RAM y otra dos máquinas equipadas con 2 GB de RAM. Cada una de las máquinas dispone de 10 GB de almacenamiento en disco duro.
- 500 GB de almacenamiento compartido vía NFS, donde se almacenaran todos los ficheros de trabajo de Matterhorn, tanto los ficheros temporales, como los de distribución de streaming.

El proceso de instalación y configuración se ha llevado a cabo siguiendo los pasos que se detallan a continuación:

En primer lugar se procede a instalar las herramientas necesarias para el funcionamiento del sistema en las 4 máquinas, en este caso las herramientas *Subversion*, *Máquina Virtual de Java* (JVM) y *Apache Maven 2*, así como la plataforma OSGI, Apache Félix.

El sistema requiere el uso de un gestor de base de datos (**SGDB**), y proporciona dos opciones: MySQL o PostgreSQL. Se ha optado por la primera de las opciones, MySQL, ya que es la opción recomendaba en la página del proyecto.

Tras copiar los ficheros necesarios a sus rutas correspondientes se descarga mediante Subversion la versión de Matterhorn que se va a utilizar.

El siguiente paso es definir las variables de entorno para la compilación.

Posteriormente se decide compilar e instalar Matterhorn utilizando para equipo un perfil distinto.

Los perfiles que se han definido para cada equipo del proyecto se describen en la **Tabla 1**.

Admin	-Padmin,dist-stub,engage-stub,worker-stub,workspace,serviceregistry
Worker	-Pworker,serviceregistry,workspace
Enage	-Pengage,dist,serviceregistry,workspace

Tabla 1 Perfiles utilizados para cada equipo Matterhorn

Tras la instalación, es necesario configurar tanto la base de datos como los propios ficheros de configuración de Matterhorn [Ver **Anexo 1**], operación necesaria para que los 4 servidores puedan interactuar entre ellos, y puedan localizar todos los ficheros necesarios para el procesamiento de los vídeos, **FFMPEG**, **Qtsubtile** (Ver **Glosario**), **MediaInfo** (Ver **pag. 112**). Para generar la base de datos, hasta la versión 1.1 incluida se ha utilizado el script que facilita Matterhorn, con la estructura que dicha base de datos debe tener. A partir de la versión 1.2, este script no es necesario, ya que las tablas son generadas automáticamente, en función de las necesidades de los servidores.

3.5. Implementación Agente de Captura

El núcleo del capturador del Sistema Matterhorn está originalmente basado en el capturador suizo SWITCHcast Recorder. Por tanto, el funcionamiento interno es muy similar al de este último. Matterhorn utiliza las mismas librerías, opciones de configuración internas parecidas, etc., aunque se ha adaptado su uso a Matterhorn, con sus propios archivos de configuración, flujos de captura, etc.

El proceso de instalación de los agentes de captura que se quieran utilizar es en principio distinto al que se sigue en el resto de equipos.

Cada agente de captura necesita dos capturadoras, una de VGA para poder grabar las presentaciones que se vayan mostrando a través del ordenador y otra para las cámaras de grabación. También necesita una entrada de audio.

Con el agente de captura, la instalación se realiza desde un script que automatiza todo el proceso y es el encargado tanto de instalar los drivers como de configurar el sistema, así como descargar la versión de Matterhorn que se elija y compilarla para el sistema sobre el que se quiera instalar.

Posteriormente hay que configurar los archivos de configuración para su integración completa en el sistema. [Ver **Anexo 1**]

Es factible disponer de varios capturadores, en distintas salas y todos ellos conectados al propio sistema Matterhorn.

Desde el administrador ya se especificará que capturador tiene que poner en marcha y realizar la grabación.

Es posible además modificar distintos aspectos de las grabaciones, aspectos tales como el bitrate de grabación, framerate, calidad, fuente, etc. para adaptarlos a las necesidades puntuales que se requieran.

3.6. Configuración Agente de Captura

Para la correcta configuración del capturador, inicialmente hay que configurar las propiedades del capturador para que este actúe con el “core” del sistema, añadiendo las rutas necesarias para que el sistema funcione, así como configurar las URLs para que el sistema lo detecte como uno de los capturadores disponibles.

También es necesario configurar la conexión con la base de datos para que de esta forma, el capturador quede integrado totalmente en el sistema Matterhorn.

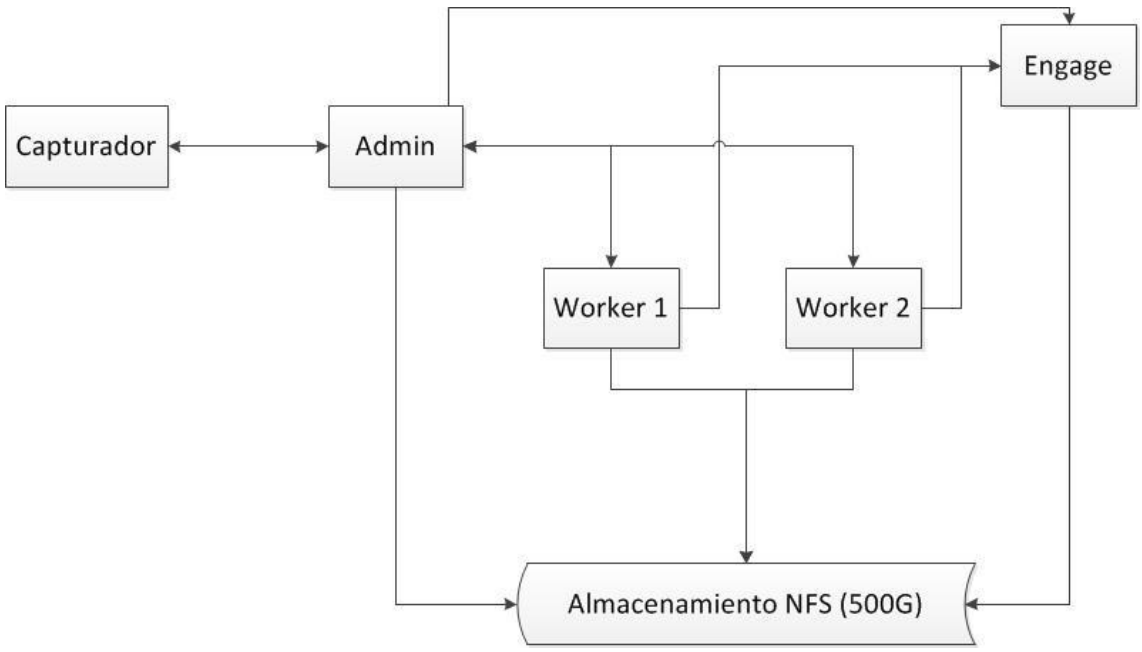


Ilustración 2 Esquema Instalación Matterhorn con capturador

Para la captura de las distintas fuentes de audio y vídeo, se utilizan las librerías Gstreamer.

Esto permite un elevado número de posibilidades de configuración, permitiendo configurar tanto la calidad de la captura, como las modificaciones que se quieran hacer al archivo en "crudo" sin necesidad de postprocesamiento posterior, como aumentar brillo, tamaño, bitrate, etc.

Para ello, es necesario modificar el fichero *org.opencastproject.org.capture.ConfiguracionManager* [Ver **Anexo 1**], para modificar todos los datos relacionados con los dispositivos de captura, como puede ser, el bitrate de las grabaciones, formato, librería de entrada, fuente de audio/vídeo, etc.

3.6.1. Captura Vídeo

Importante es mencionar que los vídeos capturados deben grabarse sin sonido, ya que en caso de que existan varias pistas de audio grabadas (la pista de audio grabada, más pistas de audio que se hayan generado junto con el vídeo) el posterior procesamiento generará errores.

- **Captura Grabación**

La captura de la grabación del ponente se realiza a través de la tarjeta capturadora Hauppauge 350 WinTV (Conexión PCI). La señal se recibe a través de un cable de vídeo compuesto (conector RCA [13] Amarillo).

Para esta capturadora, los drivers van incluidos en el sistema Matterhorn. Una vez instalados, es necesario configurar la entrada por la que se recibirá la señal de vídeo.

El formato de almacenamiento del archivo capturado debe ser el formato aceptado de forma nativa por la tarjeta capturadora.

Hay que tener en cuenta que la captura tanto del vídeo como del audio, se realiza en "crudo", por tanto, la captura debe realizarse en el formato nativo que permita la tarjeta, ya que de esta forma no se dedican recursos del capturador a realizar la conversión en el formato de los vídeos, pudiendo dedicar todos los recursos del ordenador a la captura del vídeo y del audio para que dicha operación se efectue sin problemas y sin saltos.

Matterhorn permite especificar tanto distintas "librerías"(custom_producer, hauppage), como distintas propiedades de procesamiento de vídeo, siendo todas ellas parámetros del framework GStreamer, que será el encargado de direccionar las entradas de señal hacia los ficheros de grabación.

Otras opciones que permite es definir otros valores relacionados con la captura del vídeo, opciones como el bitrate de grabación, buffer dedicado a la captura, y parámetros relacionados.

A la hora definir los valores ideales de grabación, hay que procurar encontrar un balance entre calidad y tamaño de archivo, ya que por ejemplo una diferencia del doble de bitrate, aunque en tamaños pequeños no afecte en exceso, en archivos grandes, provenientes de grabaciones largas, puede ser muy significativa la diferencia. Por ejemplo una grabación de 2 horas de duración a 1024x768 pixeles de resolución,

utilizando un bitrate de 2048 kpbs genera un archivo de 1,7 GB de tamaño, mientras que con un bitrate de 6144 kpbs, genera un archivo de aproximadamente 4 Gb.

A pesar de que hoy en día las posibilidades de almacenamiento disponibles hacen que el tamaño no sea un gran inconveniente, sí puede serlo a la hora de enviar el archivo para su procesamiento, puesto que a la hora de procesarlo requiere mucho más tiempo y recursos del servidor, y a la hora de visualizar ese vídeo la calidad que se aprecia es similar por lo que no compensa utilizar ese aumento de bitrate.

- **Captura Presentación**

La captura de la presentación que el ponente está utilizando, se realiza a través de una tarjeta capturadora que convierte la señal VGA a través de un puerto USB (Tarjetas Epiphan VGA2USB).

Para que la capturadora funcione, es necesario utilizar los binarios proporcionados por la empresa fabricante, de otra forma no es posible hacerlo funcionar ya que no existen actualmente implementaciones libres de dichos drivers.

Existen distintos modelos de capturadoras Epiphan que permiten la captura de distintas resoluciones a distintos frames. Los modelos de más coste, permiten capturar vídeo a más resolución y más framerate, tarea donde los modelos más económicos presentan más limitaciones. No obstante, puesto que el objetivo principal es la captura de diapositivas, para esta implementación es suficiente una capturadora capaz de capturar el vídeo con una resolución de 1024 x 768 pixeles a 15 imágenes por segundo, al ser una sucesión de diapositivas (presentación) el objeto de la grabación, no es un aspecto fundamental que exista fluidez en el cambio de diapositivas, momento donde se podría notar esta caída de framerate. En caso de que las necesidades del servicio requieran de la grabación de vídeos en las presentaciones, la capturadora utilizada en este proyecto, VGA2USB LR, permite grabar sin mayores inconvenientes con la fluidez necesaria, ya que las especificaciones de esta tarjeta indican que es posible grabar a una resolución de 1280x1024 pixeles a una velocidad de 51 fps, es decir el doble del framerate estándar de los vídeos.

El vídeo capturado, al igual que el audio, se graba en formato MPG (MPEG2, en el primer caso), formato con el que trabaja Matterhorn, para después poder convertirlo a cualquier otro formato requerido.

3.6.2. Captura Audio

La captura de audio la gestiona la librería **GStreamer** a través de los pipeline diseñados para tal efecto, de manera similar al proceso de captura de vídeo.

Por encima de esta librería, puede ir cualquier sistema de sonido que interactúe correctamente con **GStreamer**.

En el escenario de instalación de Matterhorn con un sistema Linux, son 2 los posibles sistemas de audio a utilizar, en función del sistema operativo sobre el que se haya instalado Matterhorn: PulseAudio y ALSA.

En este proyecto, se han probado los dos sistemas, PulseAudio, cuando el capturador corría sobre Ubuntu Desktop, y ALSA, cuando al capturador se le instaló Ubuntu Server.

Cada uno de esos sistemas requiere una pequeña configuración para asegurar el correcto funcionamiento de la grabación de audio.

- **PulseAudio**

PulseAudio es un nuevo sistema gestor de audio para Linux, escrito desde cero, para mejorar las limitaciones de otros sistemas anteriores como Alsa, como la reproducción de varias fuentes de sonidos (programas al tiempo), etc.

PulseAudio permite tanto la reproducción como el control del sonido generado por distintos programas de forma simultánea.

A la hora de aplicarlo al sistema Matterhorn, a pesar de que se trata de cambiar una línea de configuración, el proceso generado fue más extenso, debido a que PulseAudio,

por defecto y de forma recomendada, es ejecutado por cada usuario y cada instancia de las X, es decir, el servicio se inicia cada vez que un usuario inicia sesión.

Como se ha mencionado anteriormente, el servicio que proporciona el sistema Matterhorn, se inicia mediante un usuario diferente, en este caso el usuario "Matterhorn".

Al tratar de realizar una grabación, se produce el intento de conexión del sistema Matterhorn al servidor PulseAudio, pero como este no está iniciado, por lo menos para el usuario "Matterhorn, la conexión es rechazada y la grabación de audio no se puede realizar.

La forma de solucionar esta circunstancia es permitir al usuario Matterhorn el acceso al servidor PulseAudio.

El recurso utilizado ha sido configurar PulseAudio en modo "demonio" (daemon) del sistema, para que de esta forma, cuando se arranque el equipo, el servicio esté disponible para todos los usuarios y no haya problemas a la hora de que Matterhorn quiera acceder al proceso PulseAudio para grabar el sonido.

- **Alsa**

El sistema **ALSA** (Advanced Linux Sound Architecture), es el componente del núcleo de Linux encargado de la gestión del sonido del sistema.

Se trata de un sistema ampliamente utilizado en infinidad de distribuciones Linux, aunque debido a su antigüedad y limitaciones, cada vez es más frecuente que en distribuciones de escritorio de Linux sea sustituido en el nivel más alto por PulseAudio, aunque ALSA siga operativo por debajo de este.

Para los despliegues del capturador en versiones servidor es recomendable su utilización puesto que su configuración no entraña dificultad alguna y no presenta incompatibilidades a la hora de realizar la grabación.

El único punto que hay que revisar al utilizar ALSA, es que la captura de audio se encuentre activada en la tarjeta que se vaya a utilizar para dicha operación, para ello

es necesario configurar el volumen del dispositivo de captura, a través de la aplicación `alsamixer`.

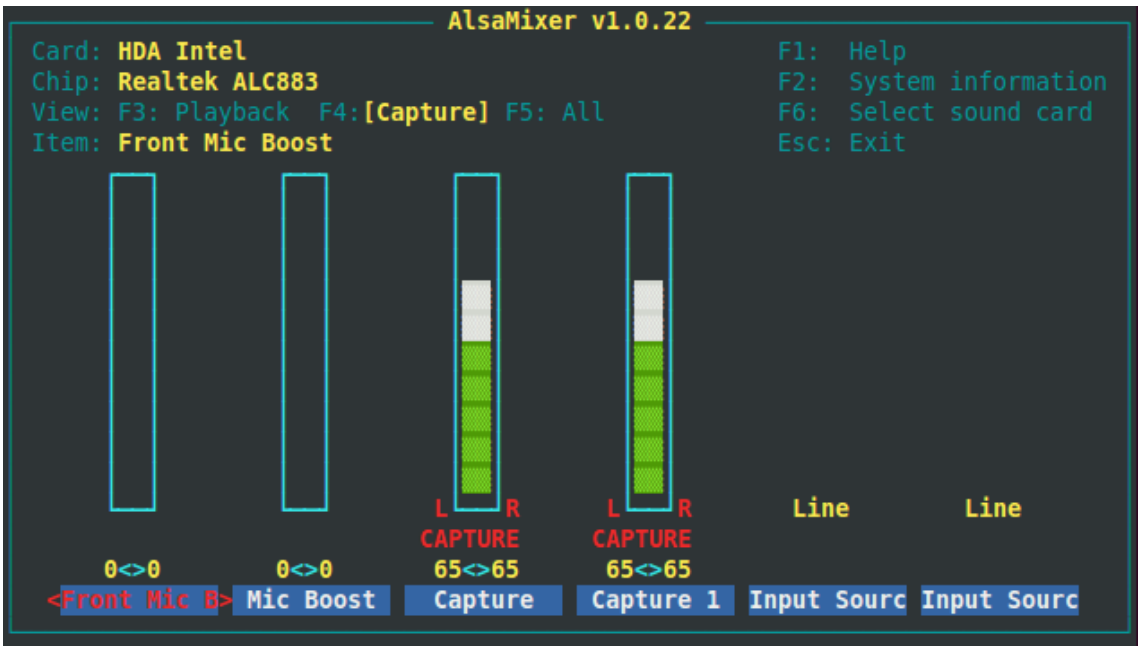


Ilustración 3 Configuración Volumen Captura ALSA (AlsaMixer)

Una vez configurado el volumen de captura, no es necesario modificar ningún otro elemento para el correcto funcionamiento de la captura de audio por ALSA

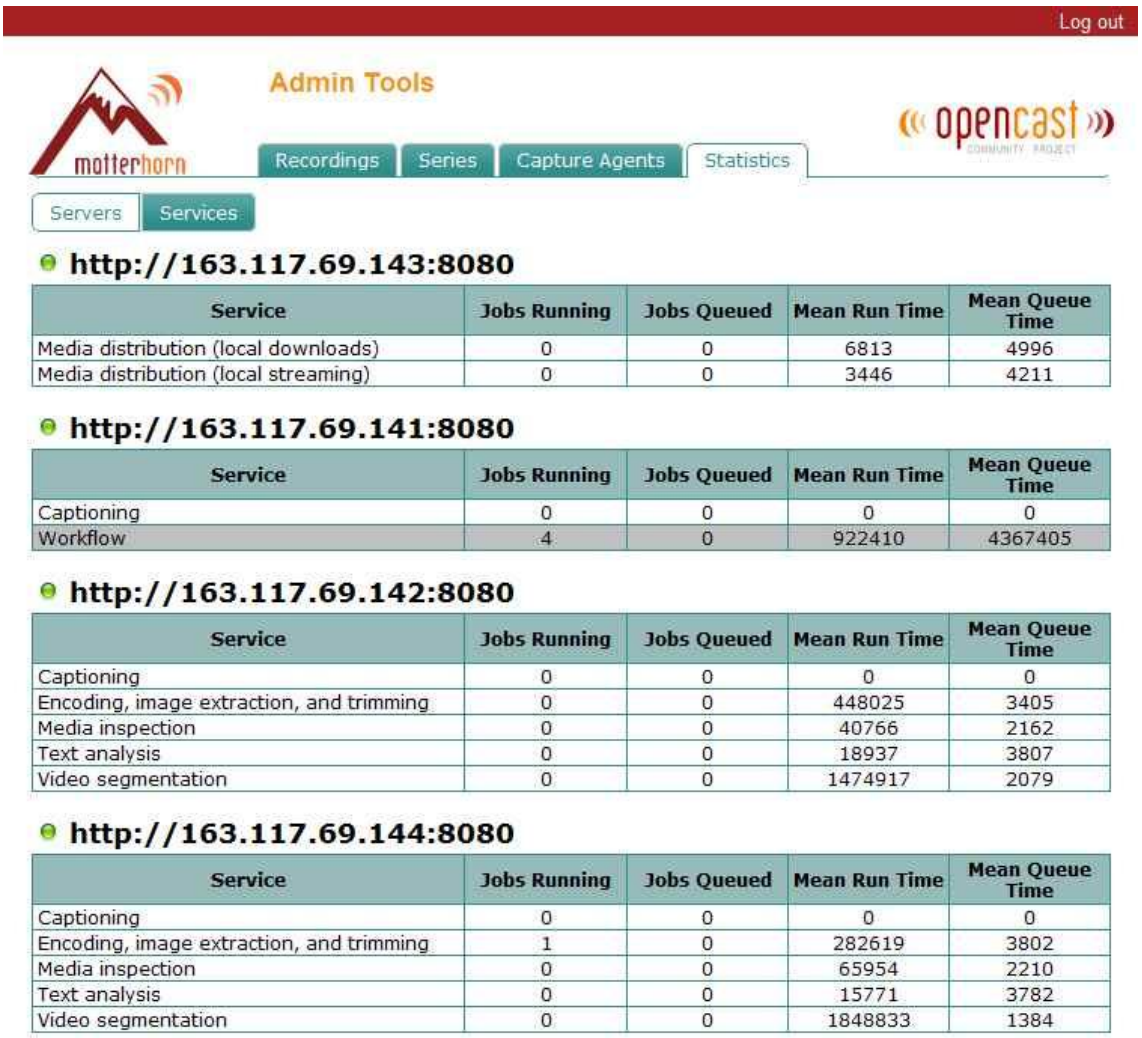
3.7. Servidor Administrativo (Admin)

Opencast Matterhorn requiere de un *Core*, un servidor que actúe como centro de control del sistema.

Este *Core* es el centro administrativo del sistema. Es tanto el encargado de mantener registrados todos los servicios y servidores del sistema, como de ejercer la gestión de las series, y lo más importante, de los Workflows de trabajo.

Esto implica que a pesar de que son los *Workers* los encargados de procesar los vídeos, la orden para que esta operación se realice proviene de este Admin.

El servidor *Admin* mantiene todos los servicios del sistema bajo control, pudiendo detectar, cuando se han añadido otros servidores y qué función realizan, para así poder distribuir la carga del sistema.



http://163.117.69.143:8080

Service	Jobs Running	Jobs Queued	Mean Run Time	Mean Queue Time
Media distribution (local downloads)	0	0	6813	4996
Media distribution (local streaming)	0	0	3446	4211

http://163.117.69.141:8080

Service	Jobs Running	Jobs Queued	Mean Run Time	Mean Queue Time
Captioning	0	0	0	0
Workflow	4	0	922410	4367405

http://163.117.69.142:8080

Service	Jobs Running	Jobs Queued	Mean Run Time	Mean Queue Time
Captioning	0	0	0	0
Encoding, image extraction, and trimming	0	0	448025	3405
Media inspection	0	0	40766	2162
Text analysis	0	0	18937	3807
Video segmentation	0	0	1474917	2079

http://163.117.69.144:8080

Service	Jobs Running	Jobs Queued	Mean Run Time	Mean Queue Time
Captioning	0	0	0	0
Encoding, image extraction, and trimming	1	0	282619	3802
Media inspection	0	0	65954	2210
Text analysis	0	0	15771	3782
Video segmentation	0	0	1848833	1384

Ilustración 4 Estadísticas de servidores de Matterhorn registrados

Es en este equipo desde donde se programan las grabaciones, operación tras la cual se almacenan los datos del capturador que debe realizar la grabación, así como el Workflow con el que se procesará la grabación.

Este servidor también proporciona el panel de control de las grabaciones realizadas, permitiendo ver la información acerca de las mismas, y gestionar aquellas que se encuentren en espera o programadas.

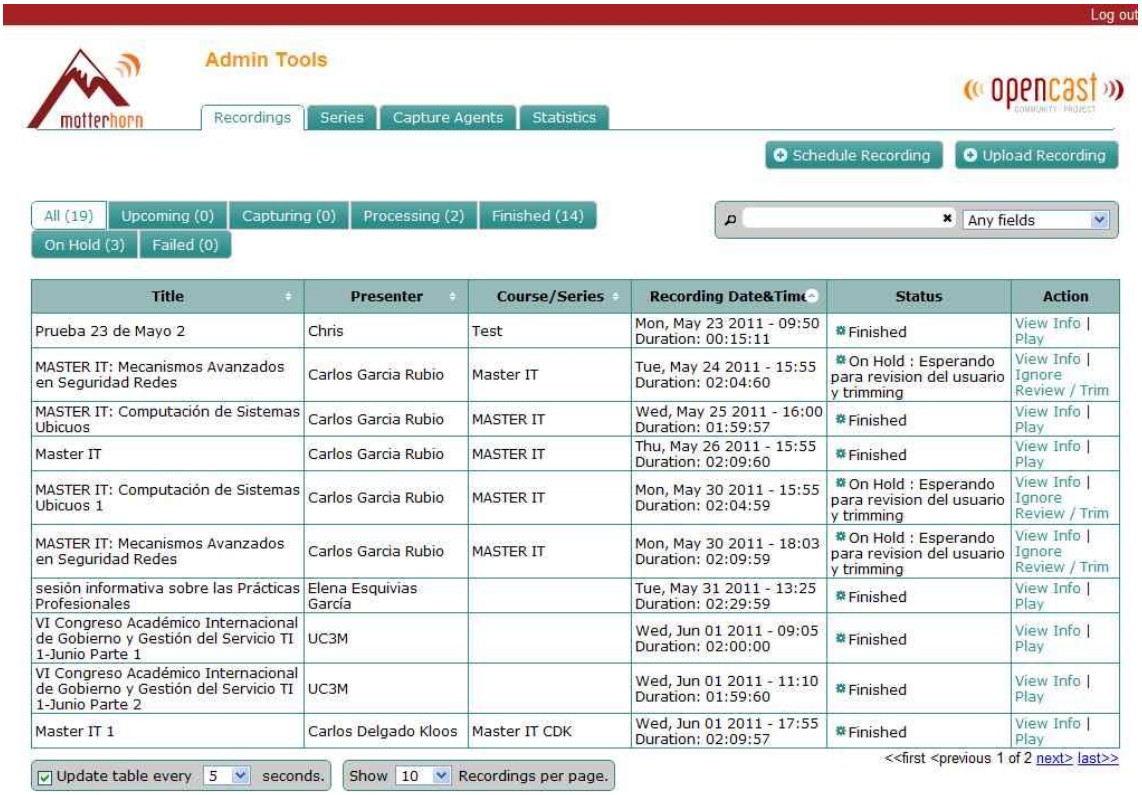


Ilustración 5 Control de grabaciones en Matterhorn

Para el correcto funcionamiento del sistema, es necesario que el resto de servidores tengan apuntada correctamente la URL del *Admin*, como el usuario y contraseña, para la correcta comunicación de los servidores, que deberá ser la misma tanto en *Admin* como en el resto de servidores de Matterhorn, capturadores incluidos. De esta forma, se registrarán correctamente todos los elementos para el funcionamiento del sistema.

3.8. Configuración procesamiento de vídeos (Worker)

Para procesar los vídeos, se utiliza **FFmpeg** como herramienta principal de codificación, no obstante, el sistema es compatible con otros codificadores como puede ser **MEncoder** o incluso Quicktime, de Apple.

También se pueden incorporar los plugins que sean necesarios para su proceso en cualquier otro formato, ya sea **H.264**, **WebM**, **Xvid**, etc.

Para la visualización en la propia plataforma, se pueden utilizar contenedores flash como **FLV** o **F4V**. Dentro de estos contenedores se puede codificar el vídeo de la forma que se desee, teniendo en cuenta tanto la calidad del vídeo en cuestión, como los equipos en los que se tenga previsto reproducir la grabación. Hay que establecer una codificación que proporcione calidad en el visionado del vídeo y tambien su retransmisión por streaming.

Para las pruebas con el contenedor F4V se utiliza como codificador la librería x264 (implementación libre del formato H.264), pero utilizando esta librería se tarda mucho tiempo en codificar los vídeos y además el reproductor no es capaz de servir correctamente los vídeos ya que presentan incompatibilidades entre el servidor de streming utilizado, el reproductor y los archivos F4V, por lo que se adopta la decisión de volver a utilizar el contenedor FLV.

La codificación con el contenedor FLV se realiza utilizando el códec H.263 para el vídeo y el códec mp3 para el audio, que proporciona una buena relación calidad/velocidad de streaming a la hora de su distribución.

En cuanto a la codificación para la descarga de vídeo, se adopta la decisión de que los vídeos tengan la opción de descargar sólo la grabación de la señal VGA, vídeo que dispone también de la señal de audio incorporada.

Esta decisión se adopta porque se considera interesante poder tener almacenado en un medio local la presentación utilizada, incluida la locución relacionada, aspecto que por ejemplo no tendría mucho sentido en caso de disponer solo de la opción del presentador hablando, sin referencia alguna de la materia tratada.

Para este archivo, tras las pruebas iniciales realizadas con parámetros por defecto (vídeos generados en contendor AVI), se comienza a investigar codificaciones con formatos de vídeo más actuales como pueden ser H.264 o el formato WebM.

Más adelante se elimina la creación del archivo WebM, por no estimarlo necesario para el actual desarrollo del proyecto y se comienzan a codificar los vídeos de las grabaciones en formato H.264 y contenedor mp4. Este último formato proporciona

unos vídeos con buena calidad y tamaño aceptable, además de ser el utilizado por ArcaMM para los vídeos descargables.

3.8.1. Perfiles de Codificación

Los perfiles de codificación a utilizar se encuentran dentro de la carpeta de configuración de Matterhorn (`<ruta que corresponda>/conf/encoding/`), en los archivos *.properties. [Ver ejemplo en **Anexo 1**]

Existen distintos ficheros que presentan distintos perfiles de codificación, ya sea para generar vídeos temporales para trabajo, para la reproducción, para su distribución a través de los feeds, para la generación de miniaturas, etc.

Estas configuraciones de las propiedades son las requeridas en las operaciones que realizan los distintos workflows que se emplean en las codificaciones.

Cada workflow dispone de diversas operaciones y realiza distintas codificaciones en función del perfil de codificación que se haya establecido previamente.

Por ejemplo, un procesado de vídeo estándar utiliza perfiles para mezclar las pistas audio y vídeo, generando un nuevo archivo con ambas pistas, tras lo que genera un vídeo para vista previa, trimming del vídeo, generación de imágenes jpg para segmentos y para el reproductor, codificación de vídeo para el reproductor y codificación de vídeo para su difusión a través de los feeds.

Las propiedades necesarias para configurar los distintos perfiles que se quiere utilizar son las siguientes:

Propiedad	**Funcionalidad**
profile.<nperfil>.name:	Nombre del perfil. Se utiliza a la hora de configurar los workflows para referenciar a estos perfiles
profile.<nperfil>.output:	Indica el tipo de salida del archivo generado. Los posibles valores son: [audio|visual|stream|image]
profile.<nperfil>.suffix:	Extensión que se añadirá el archivo generado. Debe

	ir en función de la codificación usada para que funcione correctamente
profile.<nperfil>.mimetype:	El mime type del archivo generado. Debe ir acorde a la codificación del archivo
profile.<nperfil>.input:	Tipo de archivo que se requiere para su procesado. Los posibles valores son: [audio|visual|stream|image]
profile.<nperfil>.ffmpeg.command:	Comando de ejecución ffmpeg. Este comando codifica el vídeo de entrada con el formato de salida, usando los parámetros que se especifiquen, (los valores de codificación del vídeo y también del audio, o solo comando para generación de imágenes)

Tabla 2 Propiedades para las codificaciones de los workflows

3.8.2. Separación de diapositivas (Vídeo Segmentación)

Matterhorn permite realizar una segmentación de la grabación VGA, esta operación es útil para poder acceder directamente a la sección que interese del vídeo, en función de la diapositiva que interese visualizar.

Cuando se ha definido previamente la operación de vídeo segmentación, Matterhorn utiliza para el análisis de vídeo la señal VGA en formato MPEG (para evitar problemas de compatibilidad con los diversos codecs, el proyecto Opencast decidió que en esta operación sólo se utilizara el formato MPEG).

Una “limitación” del sistema es que sólo permite reconocer un segmento durante los últimos 10 segundos de grabación, por tanto, cuando se detecta un nuevo segmento, no detectará ningún otro durante los siguientes diez segundos.

Esto permitirá que cuando se ejecute un avance o retroceso rápido de las transparencias, con el fin de localizar alguna en concreto, no se analice toda la búsqueda, sino solamente el resultado, una vez que sea localizada la transparencia en cuestión.

El algoritmo utilizado, (el algoritmo Canny Edge para detección de bordes) analiza la grabación segundo a segundo, creando una imagen reducida, utilizando el algoritmo antes mencionado y comparando la misma con la del segundo siguiente.

En caso de que las imágenes analizadas sean idénticas (por idénticas se entenderá que presentan un parecido alto, sin mucha variación, similar a la que se pueda producir por el movimiento del puntero del ratón), no realiza operación alguna, en caso contrario, la marca como nueva transparencia.

La imagen reducida es una imagen bicolor (blanco y negro) en la que se han generado los bordes de los textos que ahí aparecen, por tanto se tiene una imagen donde resulta muy fácil averiguar los pixeles cambiados.

Se realiza entonces una comparación con la siguiente imagen generada y en caso de que se detecte que el número de pixeles modificados es, por defecto, igual o superior a un 5%, la marca como nueva transparencia.

Ilustración 6 Ejemplo de Reducción de Imagen mediante el Algoritmo Canny Edge

Así mismo, con el fin de estabilizar la marca de un nuevo segmento, es necesario que aparezcan en el análisis 5 imágenes iguales para que se marque como un segmento (es decir, puesto que para el análisis se utiliza una imagen por segundo, es necesario que la imagen actual de la presentación se mantenga, durante por lo menos 5 segundos).

Estos valores son editables en el fichero *org.opencastproject.analysis.vsegmenter.VídeoSegmenter.properties* [Ver **Anexo 1**], para poder afinar más o menos la citada segmentación o poder dotarla de mayor o menor sensibilidad.

Al tratarse de un valor general para todo el sistema, no es recomendable modificarlo con frecuencia.

3.8.3. Reconocimiento de Texto

Cuando se haya especificado en el workflow que se realice el reconocimiento de texto, se analiza cada una de las diapositivas obtenidas anteriormente durante la separación de las mismas.

Una vez realizadoa esta acción, se analiza cada una de las transparencias, utilizando un sistema OCR [1]. En este caso se utiliza el programa **Tesseract**, una librería que utilizando imágenes con palabras y una amplia base de datos con diversas palabras, coeficientes de aparición, etc., puede llegar a reconocer los textos que aparecen en una imagen, un vídeo, etc.

Una vez obtenidas estas posibles palabras, se comienza a realizar un contraste con la base de datos con el fin de seleccionar la palabra adecuada en función de su aparición.

En la versión actual de Matterhorn el sistema aun no está todo lo depurado que debiera, sobre todo para el idioma castellano ya que aparecen demasiados errores en los textos reconocidos, mostrando caracteres extraños, etc.

Por esta causa se ha desactivado el sistema de reconocimiento de texto en la implementación actual.

3.8.4. Trimming (Recorte del vídeo)

Desde la versión 1.1 Matterhorn permite el recorte de los vídeos generados antes de su procesamiento.

Esta funcionalidad permite ajustar la duración del vídeo cuando la grabación tenga una duración inferior a lo esperado, así se evita tener una parte del vídeo vacía, sin acto, presentación ni recursos.

Este proceso se realiza después de haber especificado esta opción a la hora de planificar la grabación. Esta operación que aparece junto a la opción de versión previa del vídeo, codificada a poca calidad, que será suficiente para ver por qué sección se acorta la grabación.

En esta vista previa, se dispone de los dos vídeos, además del audio grabado, permitiendo además la modificación de los metadatos de cada vídeo.

Una vez seleccionados los puntos de recorte, continúa el procesamiento de los vídeos, en base a ese vídeo recortado.

También se puede aprovechar este momento para modificar los metadatos del vídeo que no se hubieran añadido anteriormente, como la fecha correcta, título, autor, etc.

3.8.5. **Subtitulado**

Cuando se vaya a procesar un vídeo, es posible, en la configuración de la grabación decidir si se va a querer adjuntar a la grabación un fichero de subtítulos.

Una vez realizada la grabación e iniciado el procesamiento del vídeo, aparecerá la opción de subir un fichero con el subtítulo del vídeo después de ejecutar la función "trimming", cuando previamente se hubiera seleccionado, o aparecerá al principio de la operación de procesamiento, en caso de no haber seleccionado la anterior opción.

Estos subtítulos son del tipo "Closed Captions", es decir, su visualización dependerá de la voluntad del usuario, que podrá decidir si quiere verlos o no.

El formato que acepta Matterhorn para subtítulos es el formato "Timed Text Authoring Format 1.0 – Distribution Format Exchange Profile (TTAF DFXP)". Este formato, DFXP, es una recomendación del W3C y está basado en el lenguaje XML. Para su generación se puede utilizar cualquier editor de subtítulos que permita exportar o bien a DFXP directamente, o bien a otro formato como puede ser SubRip (*.srt), ya que Matterhorn proporciona también un servicio para la conversión de subtítulos.

Una vez incorporado el subtítulo en formato DFXP al vídeo a procesar y una vez finalizado el procesamiento del mismo, el reproductor de Matterhorn proporciona la opción de mostrar u ocultar los subtítulos que se hayan añadido.

Esta funcionalidad presenta una importante limitación en su utilidad, Matterhorn no proporciona un sistema para generar estos subtítulos desde el propio sistema, se debe proceder a su generación de forma externa, con la dificultad añadida por el propio diseño de los workflows de trabajo de Matterhorn, que resulta difícil generar, aunque sea de forma externa, un subtitulado correcto para casi cualquier vídeo.

La causa es imputable a la función “trimming” o más concretamente a su uso.

A continuación se detallan los pasos a seguir para realizar una grabación con subtítulos.

Una vez finalizada correctamente una grabación se puede decidir realizar, mediante un programa externo, el subtitulado del vídeo, teniendo como vídeo base, la vista previa que proporciona la opción “trimming”, y en base a los datos obtenidos se puede realizar el “timming” del vídeo, subtitulando la exposición del ponente hasta finalizar el proceso.

En esa misma opción de trimming en la que se encuentra el vídeo a procesar, se realiza el corte del vídeo por delante, eliminando el tiempo muerto hasta el inicio del ponente. Pero el subtítulo que en este caso se ha generado, no serviría puesto que estaría descompensado respecto con la duración del vídeo final.

Una posible solución sería crear el subtítulo teniendo en cuenta el tiempo de vídeo que se va a eliminar. De esta forma sí que se obtendrá un subtítulo correcto y acorde a la duración real de la grabación.

Hubo un proyecto de subtitulado para Matterhon llamada OpenCaps, que facilitaba el proceso de subtitulado de vídeos, pudiendo elegir la grabación deseada y generar unos subtítulos acordes a la misma, tras lo cual el subtítulo se añadía a la grabación de Matterhorn.

No obstante, este proyecto se abandonó en una fase muy temprana, siendo actualmente incompatible su utilización con la versión actual de Matterhorn.

3.9. Servidor de Streaming (Engage)

A pesar de que Matterhorn puede distribuir el contenido sin necesidad de ningún plugin o add-on, el rendimiento es mucho mejor en caso de disponer de alguno de estos. En este caso, tal y como se recomienda desde el propio proyecto, Matterhorn se sirve del sistema de streaming Red5.

Matterhorn permite, gracias a su arquitectura modular utilizar otros servidores de streaming, como puede ser la solución ofrecida por Adobe para este cometido, Flash Media Server.

Sin embargo dado que el soporte oficial de Matterhorn utiliza Red5, se adopta ña decisión de mantener dicho servidor.

Con esta opción, se sigue manteniendo un sistema LCS lo más “libre” posible, utilizando la menor cantidad posible de software privativo.

El proceso requiere instalar el servidor de streaming, y posteriormente, compilar e instalar la aplicación que permitirá que el Engage de Matterhorn interactue con el servidor de streamning para servir los vídeos.

3.9.1. Reproductor

El reproductor de Matterhorn es un reproductor basado en tecnología Flash.

Este reproductor permite visualizar simultáneamente, en multistream, los dos vídeos grabados (vídeo del presentador y vídeo de la señal VGA, **Ilustración** 7, **(1)** y **(2)**) en sincronía con el audio.

La URL de los vídeos es de la forma:

```
http://<direccion o IP>:<puerto>/engage/ui/watch.html?ID=<ID del vídeo>
```

En este reproductor se encuentran los controles de reproducción estándar, reproducción, pausa, avance, retroceso y los típicos botones de pasar archivo (**Ilustración** 7, **(3)**) que en este reproductor cumplen la función de avanzar el vídeo hasta la siguiente diapositiva.

El reproductor incorpora un contador de duración del vídeo donde se puede introducir el momento exacto al que se quiere acceder en cada momento (**Ilustración 7**, **(4)**).

También dispone de una barra de progreso de la reproducción, con la que se puede avanzar y donde se encuentran marcados los segmentos en los que se ha dividido la grabación, en función de las diapositivas disponibles. (**Ilustración 7**, **(5)**)

El acceso a la sección del vídeo que interese se realiza pulsando directamente en la sección de la barra de reproducción que se quiera seleccionar. (**Ilustración 7**, **(6)**)

Como los vídeos que se reproducen son independientes entre sí, el reproductor estándar permite modificar el tamaño de cada vídeo, para así poder centrarse en el material disponible que interese al usuario.

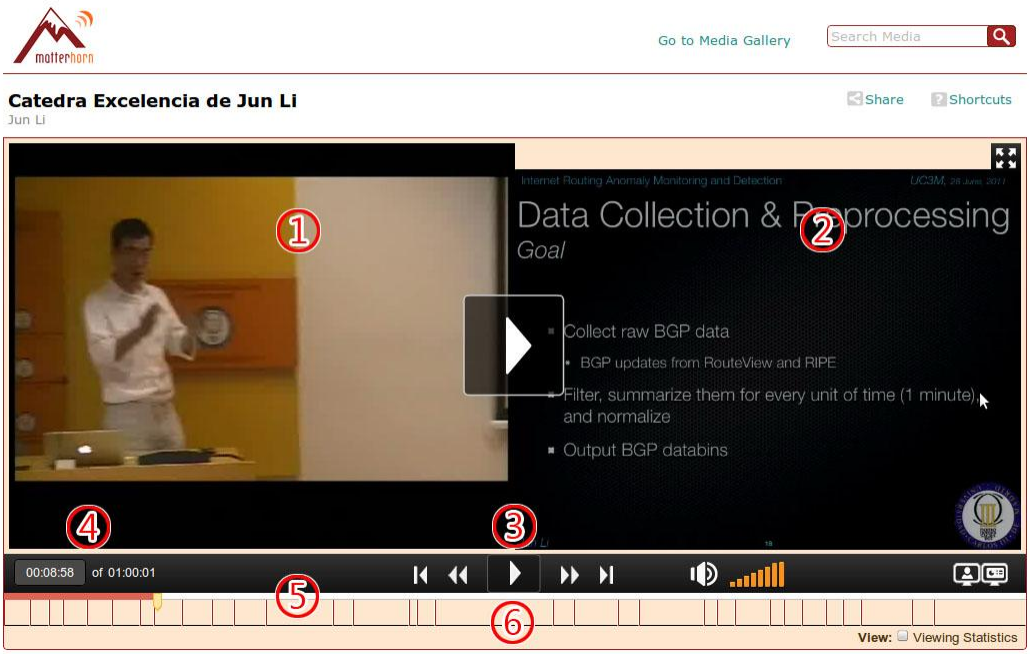


Ilustración 7 Imagen Reproductor de vídeo

Otros elementos disponibles en el reproductor se encuentran en las pestañas de la parte inferior y que se describen a continuación.

- Pestaña "**Description":** En la primera pestaña (ver **Ilustración 8**) se muestran datos generales sobre el vídeo, idioma, autor, etc.

Ilustración 8 Pestaña "Description" del reproductor

- Pestaña "**Segments":** En la segunda pestaña (ver **Ilustración 9**) se muestran las imágenes de los segmentos generados del vídeo, que permiten acceder directamente a la sección del vídeo que se desee.

Ilustración 9 Pestaña "Segments" del reproductor

- Pestaña "**Segments Text":** En la tercera pestaña (ver **Ilustración 10**), estarían las miniaturas generadas y deberían aparecer también los textos de dichas diapositivas, una vez se haya realizado el reconocimiento de caracteres. No obstante, en la versión 1.2 esta funcionalidad se encuentra desactivada por problemas con el OCR (no reconoce correctamente el texto). Por tanto tampoco tiene funcionalidad alguna el recuadro de "Search this recording" (ver **Ilustración 10**) que permitirá, una vez que la funcionalidad se encuentre activada y en perfecto funcionamiento, buscar la sección del vídeo donde aparezca el texto buscado.

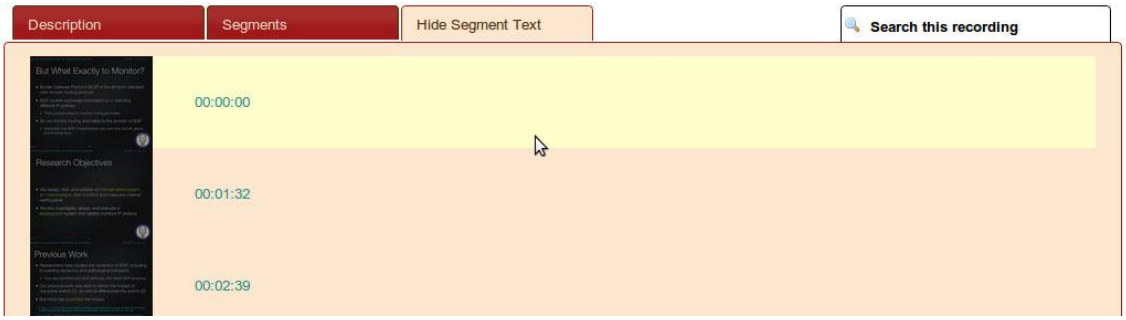


Ilustración 10 Pestaña "Segments Text" del reproductor

- # Reproductor Embebido

Existe también la posibilidad de disponer de un reproductor embebido para visualizar las grabaciones de Matterhorn.

Este reproductor es ideal para su inserción en cualquier página o blog, ya que a diferencia del reproductor de Matterhorn, no ocupa toda la pantalla, sino un pequeño espacio que es posible elegir entre varios tamaños.

Este reproductor no permite la visualización en multistream de las grabaciones. Únicamente permite mostrar una de las dos fuentes de vídeo, aunque se puede alternar una fuente con otra en cualquier momento de la reproducción (Ilustración **11, (1)**)

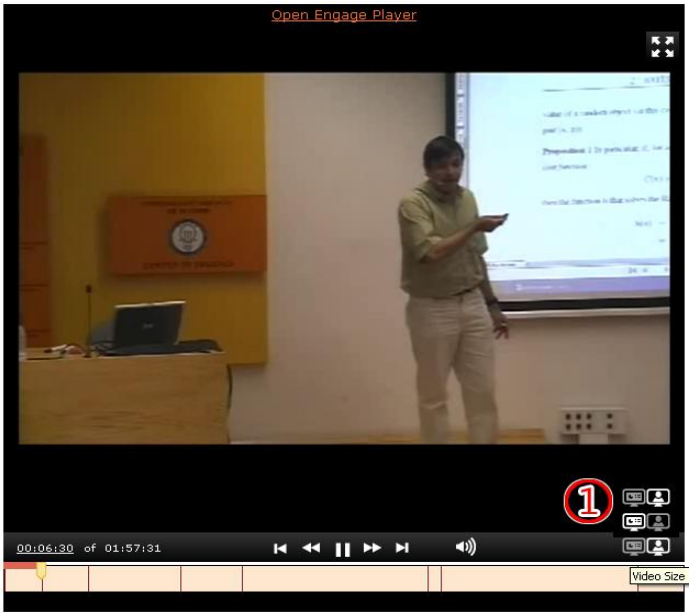


Ilustración 11 Vídeo Presentador Reproductor Embebido

Proporciona además en la misma ventana un enlace para ver la grabación en el reproductor completo de Matterhorn (**Ilustración 12, (2)**).

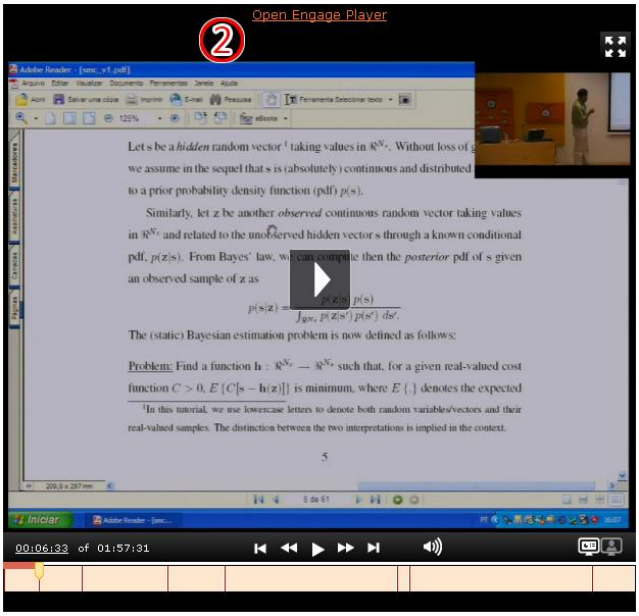


Ilustración 12 Vídeo VGA Reproductor Embebido

Este reproductor mantiene alguna de las características de la versión completa, como mostrar subtítulos, etc., aunque sacrifica otras opciones como puedan ser la anteriormente mencionada visualización en multistream o el acceso directo por diapositivas (solo se visualizan los segmentos, no las diapositivas donde se encuentran).

Matterhorn permite la opción, desde la ventana del reproductor, de generar el código necesario para poder embeber el reproductor en el sitio que se desee.

Este reproductor se inserta en cualquier página por medio de un "*iframe*", donde a pesar de los valores que proporciona Matterhorn para las etiquetas HTML, los elementos necesarios son la dirección (aquí es de la forma `http://<direccion o IP>:<puerto>/engage/ui/embed.html?ID=<ID del vídeo>`), y también ek ancho y alto del *iframe*, el resto son etiquetas cuyos valores deben estar ya tratados mediante hojas de estilo en las páginas destino.

Por defecto, Matterhorn proporciona una serie de tamaños de ventana para su selección, pero siempre es posible modificar estos valores para ajustar el reproductor embebido a las necesidades concretas de la página en la cual se vaya a incluir el vídeo.

Ilustración 13 Tamaños por defecto para el reproductor embebido

Una vez seleccionado el tamaño deseado, Mattehorn proporciona un código similar al que se muestra en la **Tabla 3**. Este es el código que se agregará a la página web en la que se quiera visualizar el vídeo en el reproductor embebido.

```
<iframe src="http://163.117.69.143:8080/engage/ui/embed.html?ID=852"
style="border:0px #FFFFFF none;" name="Opencast Matterhorn Media Player"
scrolling="no" frameborder="0" marginheight="0px" marginwidth="0px"
width="540" height="532"></iframe>
```

Tabla 3 Código para incrustar el reproductor de Matterhorn

3.9.2. Feeds

El servidor engage de Matterhorn, proporciona otra opción de distribución de contenidos: feeds con la información de las grabaciones realizadas.

Un feed es un medio de redifusión de contenido web. Se utiliza para suministrar a los suscriptores información actualizada acerca del contenido de una web, especialmente con las últimas novedades que se hayan generado.

La información proporcionada a través de los feeds, recoge entre otros datos, información relativa al vídeo, autor, fecha, título, serie, etc.

Matterhorn permite definir distintos tipos de feeds, con los que se pueden realizar distintas búsquedas por cada feed. Por ejemplo, se puede definir un feed para que muestre las grabaciones de una determinada serie, o bien por un determinado autor, o que muestre solamente las últimas grabaciones realizadas, etc.

La configuración de los feeds y las distintas opciones disponibles se encuentra documentada en la página web del proyecto Matterhorn,

Los feeds generados por Matterhorn son del tipo ATOM y tipo RSS.

En los dos feeds se disponen de los datos típicos de RSS, datos específicos del formato iTunes RSS y algunos otros datos del modelo de metadatos Dublin Core.

El objetivo de estos feeds es que se pueda utilizar directamente el feed para diversos cometidos como ingestar directamente un vídeo a iTunes o a otro sistema que reconozca los modelos de datos utilizados.

No obstante, se ha apreciado que presenta una carencia importante de datos en el feed, por lo que no se utilizará directamente para introducir las grabaciones en el sistema ArcaMM. En su lugar, se utilizará este feed para obtener determinados datos (título, ID, autor, idioma, asunto, etc), que se completarán posteriormente en la siguiente fase del proyecto, con los datos que se obtengan tras analizar los propios ficheros que se quieran ingestar.

De esta forma, habrá disponible un conjunto de metadatos de cada vídeo más completo, con los que se podrán realizar las operaciones que interesen con ellos.

3.9.3. Descarga de ficheros

Matterhorn permite de manera opcional a la hora de diseñar los workflows, la generación de un vídeo destinado a la descarga por parte del usuario.

Pueden generarse vídeos con la vista del presentador, de la señal VGA o de ambas para su posterior descarga. Esta posibilidad como se ha mencionado en la sección anterior referida al Worker, en principio sólo se va a generar un solo vídeo con la señal VGA y el audio de la grabación, siendo codificado este vídeo utilizando el códec H.264.

La ruta para descarga del fichero se encuentra en los datos del feed RSS generado, por lo que se puede utilizar directamente para su descarga o bien como se verá más

adelante, se podrá utilizar ese dato para poder añadir después el enlace de descarga en la ubicación que se desee.

3.10. Workflow

Los workflows definen el flujo de trabajo que realizará el sistema Matterhorn.

Se encuentran en la carpeta homónima "workflows".

En cada uno se incluyen tanto las órdenes de codificación, como de distribución, análisis de texto, OCR, etc.

Cada workflow consiste en un archivo XML con la siguiente estructura:

```
<definition>
<ID>(ID que se asigna al </ID>
<title>(Nombre del workflow para su fácil identificación)</title>
<description>
       (Descripción muy breve de lo que hace el workflow)
</description>
<operations>
</operations>
</definition>
```

Tabla 4 Ejemplo Estructura Workflow

Dentro de la etiqueta <operations> se definen las fases que se van realizando durante el procesamiento.

```
<operation
  ID="(ID de la operación a realizar)"
  fail-on-error="true"
    exception-handler-workflow="error"
    description="Inspeccionado el paquete">
    <configurations>
<configuration key="source-flavor"> presentation/work </configuration>
<configuration key="source-flavor"> presentation/work </configuration>
    </operation>
```

Tabla 5 Ejemplo Estructura Operación del Workflow

inspect	Comprueba que no haya fallos en el mediapackage creado.
prepare-av	Une el audio grabado con los dos vídeos creados para así tener un par de vídeos completos con los que ir trabajando.
review	En caso de haber seleccionado esta opción en el panel de control, permite que el sistema quede en espera hasta que el usuario compruebe que todo está correcto (metadatos, vídeo) y le permita al sistema completar el procesamiento.

caption	Permite que el sistema quede en espera hasta que se suban los subtítulos para la grabación en cuestión.
compose	Esta operación es la que permite procesar los vídeos en función de la codificación que se haya elegido. Esta operación es la encargada de codificar todos los vídeos, ya sea para la vista previa, versiones para el reproductor o para la descarga directa.
segment-vídeo	Analiza la grabación de VGA para encontrar cambios en la presentación y así poder dividir en secciones la grabación.
segmentpreviews	Esta operación genera, en caso de que se hayan detectado cambios en la diapositiva, las miniaturas correspondientes a la segmentación realizada.
image	Esta operación es la encargada de generar imágenes estáticas, ya sea para la galería o para la vista previa.
extract-text	Comienza el analizador de textos (OCR).
distribute download	Envía la grabación a los canales de distribución elegidos (RSS, paginas de vídeo, etc.).
distribute-streaming	Envía la grabación a los canales de distribución de streaming.
publish	Añade la grabación y sus metadatos al motor de búsqueda del sistema Matterhorn.
cleanup	Finaliza el workflow y elimina los archivos temporales generados.

Tabla 6 Operaciones Matterhorn y su ID

3.11. Versiones de Matterhorn Utilizadas y cronología de la implantación

3.11.1. Matterhorn 1.0 (Octubre 2010- Enero 2011)

Primera versión utilizada. Problemas ocasionales en la transferencia de las llamadas REST.

Los primeros intentos de instalación fueron infructuosos, ya que se disponía de unos servidores con poca memoria RAM y los valores sugeridos para la compilación no servían.

No obstante, al modificar y reducir estos valores, se pudo realizar la instalación sin mayores dificultades, aunque daba como resultado un sistema configurado por debajo de las posibilidades recomendadas.

La comunicación entre los servidores generalmente se producía de forma correcta, aunque a veces fallaba el ingestado de los MediaFile, y como consecuencia no se procesaba ninguna grabación. La solución adoptada consistía en reiniciar el sistema Matterhorn, logrando que así se comunicaran correctamente los servicios de nuevo.

Esta versión provocaba además problemas con el gestor de series y con algunos tipos de feeds.

Las características principales de este sistema, versión 1.0 son las siguientes:

- Grabación de clases en multistream.
- Posibilidad de planificar grabaciones recurrentes.
- Reconocimiento de texto en las diapositivas.
- Separación de segmentos de la presentación.

3.11.2. Matterhorn 1.0.1 (Enero 2011 – Febrero 2011)

Además de las características correspondientes a la anterior versión, esta "release" supuso un aumento significativo de la estabilidad del sistema, aunque no incorporaba ninguna funcionalidad significativa.

3.11.3. Matterhorn 1.1.x (Febrero 2011 – Abril 2011)

Esta "release" incorporaba bastantes actualizaciones, mejoraba el flujo de trabajo de los workflows, y permitía la inserción de subtítulos. Incluía también un módulo básico de edición de los vídeos grabados que permitía acortar el vídeo tanto por el principio como por el final.

<u>Operaciones mejoradas.</u>

- Mejora de la instalación de los 3rd party scripts (FFMPEG, codecs adicionales y demás herramientas externas necesarias para la ejecución de Matterhorn)
- Incorporación de la funcionalidad de balanceo de carga entre distintos servidores de Matterhorn
- Incorpora la opción de configurar cualquiera de los servidores del sistema en modo mantenimiento, provocando que ese servidor no efectué trabajo algún mientras se mantenga dicho modo.
- Mejora de la documentación del código y limpieza del código Java así como de las APIS de las llamadas REST

Herramientas Administrativas

- Edición en lote para grabaciones planificadas
- Filtrado/Búsqueda de grabaciones
- Visor de detalles de grabación y gráficos de rendimiento del procesamiento
- Herramienta para revisar y cortar la grabación al principio o al final.
- Soporte de varios *inbox* cada uno con su workflow determinado
- Sistema de monitorización de tareas y servicios de los servidores

La mejora del sistema de balanceo de carga permite una mejor distribución de la carga de trabajo cuando existen varios equipos dedicados al procesamiento de los vídeos.

En esta versión se mejora el reproductor de vídeo, permitiendo el uso de archivos F4V, también mejora la gestión de los feeds de distribución de contenido, permitiendo más posibilidades de configuración.

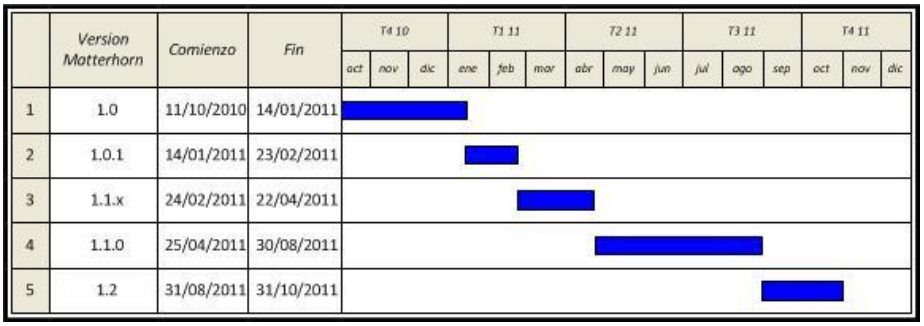
3.11.4. Matterhorn 1.1 (26 Abril 2011 – 30 Agosto 2011)

Versión definitiva de la release 1.1. Incorpora las novedades anteriormente mencionadas y añade además la estabilidad del sistema, que mejora de forma significativa, mostrándose más robusto y con menos problemas de comunicación entre los servidores

3.11.5. Matterhorn 1.2 (31 Agosto 2011)

Esta versión es la que está actualmente desplegada en la infraestructura de la UC3M, aportando las siguientes mejoras:

- Autorización por usuarios para acceso al *engage*, determinando que grupo de usuarios puede ver cada serie de grabaciones
- Integración con directorios de usuario (LDAP, etc.)
- Mejora del manejo de los metadatos de tipo Dublin Core
- Mejoras en seguridad del sistema
- Soporte para subir archivos de medios grandes (+2GB) al *inbox*
- Mejoras en la interfaz de usuario
- Solucionados algunos bugs menores que aparecían en versiones anteriores



	Version Matterhorn	Comienzo	Fin
1	1.0	11/10/2010	14/01/2011
2	1.0.1	14/01/2011	23/02/2011
3	1.1.x	24/02/2011	22/04/2011
4	1.1.0	25/04/2011	30/08/2011
5	1.2	31/08/2011	31/10/2011

Ilustración 14 Cronograma Versiones Matterhorn Utilizadas

3.12. API REST de Matterhorn

La comunicación entre los distintos nodos de Matterhorn se realiza a través de llamadas REST.

REST (Representational State Transfer) o **Transferencia de Estado Representacional** es una técnica que se aplica a los sistemas distribuidos hipermedia para intercambiar información entre ellos.

REST se sirve del protocolo HTTP para intercambiar información, encontrándose esta información generalmente en formato HTML o XML.

El concepto fundamental de REST es que todos los elementos son recursos y que deben disponer de una **URI** accesible para poder interactuar entre ellos.

A través de esta URI, un "cliente" envía una petición a través del protocolo HTTP, una llamada que, por ejemplo, añade un usuario al sistema, junto con la información que requiere transmitir, en este ejemplo seria "nombre".

Esta petición es recibida por el "servidor" junto con la información asociada con la que se debe realizar la operación (esta información debe estar en un "idioma" que entiendan cliente y servidor, por ejemplo, en XML), tras lo que realiza la operación solicitada.

Ademas de usar este tipo de llamadas de forma “oculta” al usuario, internamente, Matterhorn proporciona una interfaz desde la cual poder controlar el sistema, proporcionando API’s para cada servicio del sistema.

Estas API se encuentran accesibles desde la página principal de cada nodo, mostrando solamente aquellas que soporta dicho nodo.

Caption REST Endpoint	Servicio para conversión de formatos para los subtítulos.
Capture Agent Admin	Servicio que controla el registro de los capturadores disponibles en el sistema y también el estado de las grabaciones que realicen
Ingest REST Endpoint	Servicio que crea los mediapackages que posteriormente se procesarán, añadiendo las pistas y los datos necesarios
Runtime Information REST Endpoint	Servicio que proporciona información sobre el entorno donde se ejecuta Matterhorn y los servicios disponibles
Scheduler REST Endpoint	Servicio que crea, edita y recupera grabaciones planificadas
Series REST Endpoint	Servicio que crea, edita y recupera los datos de las series para los vídeos grabados
Service Registry REST Endpoint	Servicio que controla y gestiona el resto de servicios de Matterhorn
Workflow REST Endpoint	Servicio que lista los workflows disponibles permite su control, pudiendo iniciarlo, pausarlo, o incluso eliminarlo
Working File Repository REST Endpoint	Servicio que proporciona acceso al almacenamiento local para los procesos que lo necesiten
Composer REST Endpoint	Servicio que controla la aplicación FFMPEG, utilizada para la codificación de los vídeos
Media Inspection REST Endpoint	Servicio que extrae metadatos de los archivos generados
Text Analysis REST Endpoint	Servicio que envía un fichero de vídeo al análisis de OCR
Vídeo Segmentation REST Endpoint	Servicio que envía un vídeo al proceso de segmentación
Annotation REST Endpoint	Servicio utilizado para la gestión de anotaciones generadas por los usuarios.
Download Distribution REST Endpoint	Servicio que distribuye los mediapackages a los feeds de distribución así como al engage

Search REST Endpoint	Servicio que indexa y permite la consulta de los vídeos generados
Streaming Distribution REST Endpoint	Servicio que distribuye los mediapackages al sistema de streaming
User Tracking REST Endpoint	Servicio que permite la generación de estadísticas
Capture REST Endpoint	Servicio que controla el capturador
Confidence Monitoring REST Endpoint	Servicio que permite monitorizar el capturador,
State REST Endpoint	Servicio que muestra una visión simple del estado del capturador

Tabla 7 API's de servicios de Matterhorn

Desde la página donde se muestran estas API's es posible su manipulación, y modificación.

En dicha página se puede ver como para cada método se explican los parámetros que aceptan, los valores de retorno y otros datos relacionados.

El uso de estas APIs, entraña cierta dificultad a la hora de definir los parámetros requeridos, aunque es posible utilizarlas para extender el funcionamiento de Matterhorn. No obstante, esta tarea será objeto de trabajo posterior.

Existen otras tareas, como puede ser la eliminación de un workflow en proceso, o la eliminación de grabaciones fallidas, fácilmente realizables desde las API's.

Por ejemplo, para eliminar un workflow en proceso, tras obtener la ID de dicho workflow, es necesario acceder a la API **Workflow REST Endpoint** y donde se encuentra el método **POST /stop**.

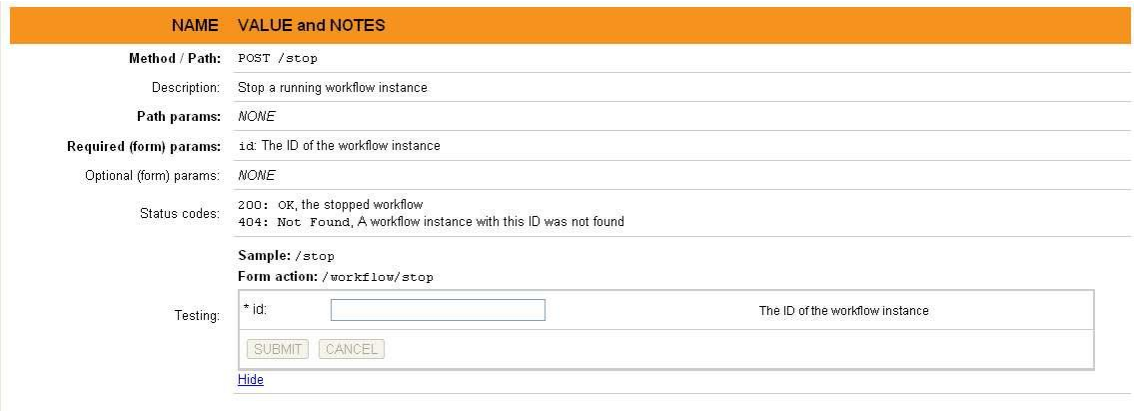


Ilustración 15 Ejemplo de Borrado de Workflow

En ese método se observa que existe un campo donde se puede introducir texto.

En dicho campo se escribirá la ID del workflow que queremos eliminar.

Una vez introducida la ID del proceso a eliminar y tras pulsar el botón correspondiente, el sistema devolverá un código que indica que la eliminación del workflow se ha realizado correctamente, o por el contrario, un código distinto que indica que se ha producido un error durante el procesado de esta acción.

3.13. Ejemplo de Grabación con Matterhorn

Para realizar una grabación estándar con Matterhorn, y hasta el momento en que se reproduce el vídeo, es necesario realizar una serie de pasos que se detallan a continuación:

1. En el Panel de Administración de Matterhorn pulsar en el botón “Schedule Recording”, para planificar una grabación

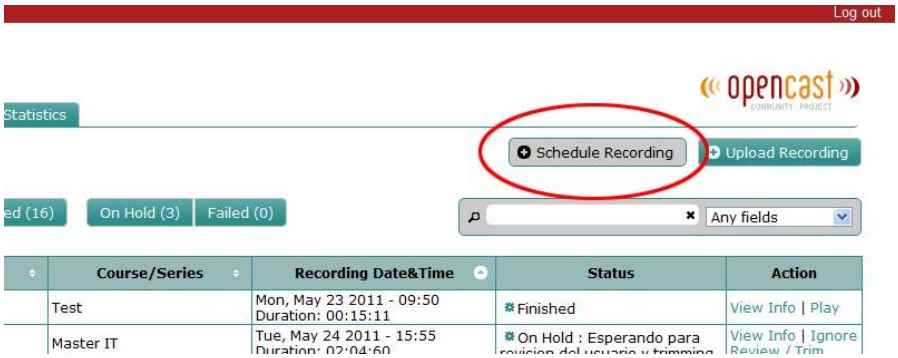


Ilustración 16 Ejemplo Grabación 1

2. En la pantalla de “Programar Grabación” tras elegir la opción “Single Recording” (también se puede programar una serie de grabaciones periódicas), hay que completar los datos relativos a la grabación (título, autores, serie. (**(1) en Ilustración 17**), Fecha de comienzo y Duración de la grabación (**(2) en Ilustración 17**), Capturador que realizará la grabación y los dispositivos que se utilizarán (**(3) en Ilustración 17**), Workflow con las operaciones que se realizarán durante el procesado de este vídeo (**(4) en Ilustración 17**) y finalmente en la opción de Pausas para control, se marcará la opción “Review / Trim before encoding” (**(5) en Ilustración 17**), que permitirá revisar la grabación y editarla antes de que se continúe con el procesamiento de la misma.

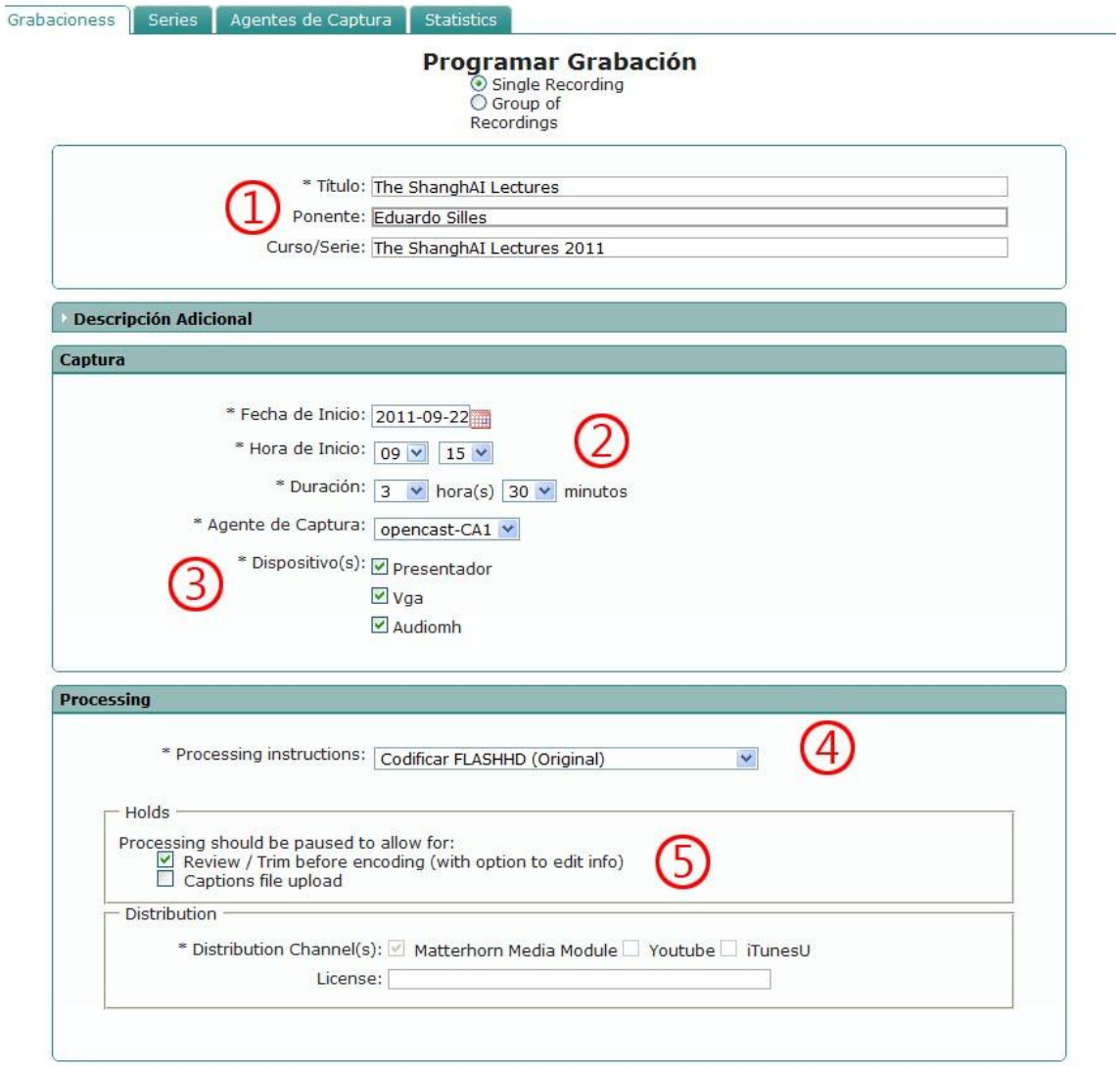


Ilustración 17 Ejemplo Grabación 2. Introducción Datos

3. Tras rellenar todos los campos necesarios y pulsar "Schedule" para finalizar la grabación, aparece una ventana que indica que la grabación se está introduciendo en el sistema.

Ilustración 18 Ejemplo Grabación 3

4. Si la grabación se ha programado correctamente, en el panel de administración, en la pestaña “Upcoming” aparecerán los datos del vídeo programado, a la espera de que llegue la hora en que se active la grabación.

Ilustración 19 Ejemplo Grabación 4

5. Cuando llega la hora programada de la grabación, el panel de control “ordena” al capturador elegido que comience la grabación.

   Durante este tiempo, aparecerá el estado de la grabación como”Capturing”

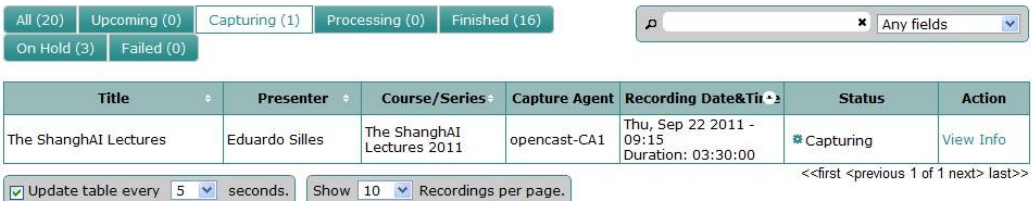


Ilustración 20 Ejemplo Grabación 5

6. Una vez que la grabación haya finalizado se iniciará su envío al núcleo de Matterhorn así como su procesado, y su estado cambiará a “Processing”

Ilustración 21 Ejemplo Grabación 6

7. Cuando se hayan generado las versiones de trabajo y la vista previa, la grabación quedará en “Hold”, permitiendo revisar y editar la duración de la grabación antes de continuar con el procesado.

Ilustración 22 Ejemplo Grabación 7

Es necesario pulsar en la opción “Review/Trim”, de esta forma se accederá al editor.

8. En esta sección, se puede acortar la duración de la grabación, especificando el punto de comienzo y el punto de final, permitiendo en caso necesario, cambiar o añadir datos de la grabación (idioma, ponentes, etc).

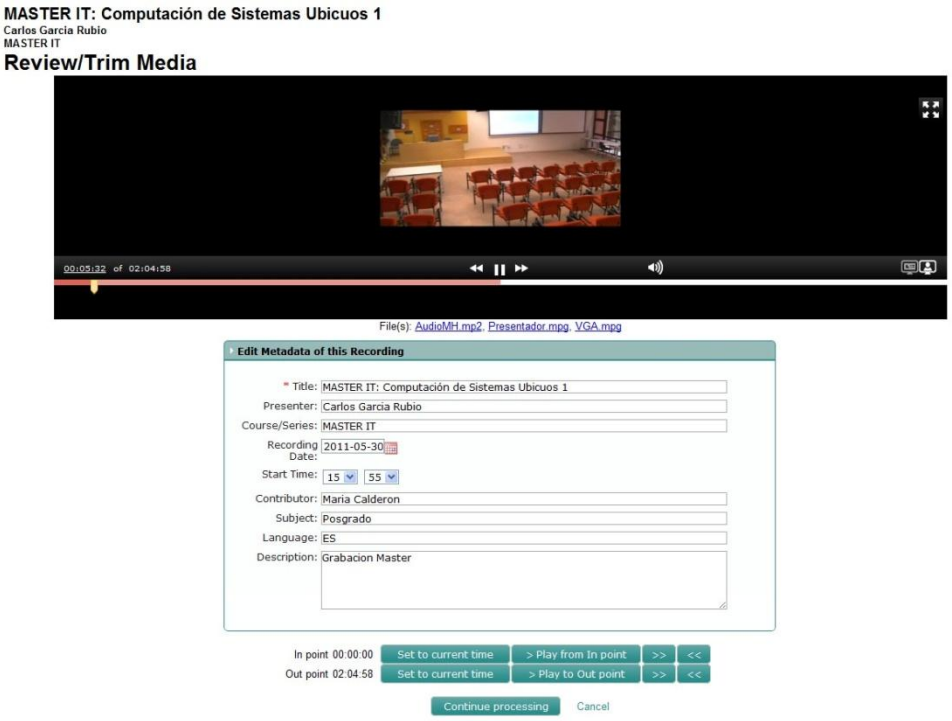


Ilustración 23 Ejemplo Grabación 8. Edición del vídeo y datos anexos

Una vez que se hayan realizado los cambios necesarios, se deberá pulsar el botón “Continue Processing”, que se encuentra en la parte inferior de la página, para continuar con el procesado del vídeo.

9. Tras realizar este paso, la grabación volverá a aparecer en la pestaña "Processing", pestaña en la que se mantendrá hasta que finalice el procesado de los vídeos.

Ilustración 24 Ejemplo Grabación 9

10. Completado el procesado y la distribución de la grabación, ésta cambiará de estado en el panel de administración y se podrá encontrar en la pestaña "Finished".

    Para ese vídeo, se podrán ver dos enlaces en la parte derecha, "View Info" y "Play".

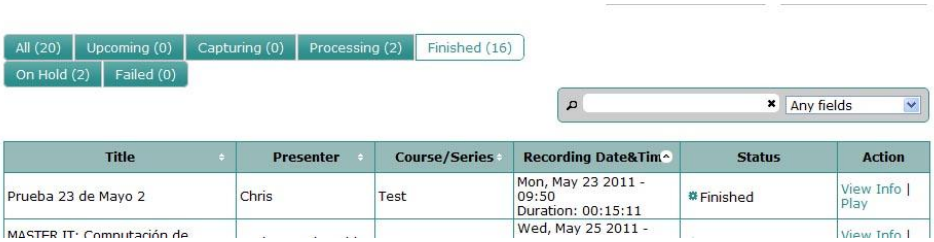


Ilustración 25 Ejemplo Grabación 10

Este último es un enlace al reproductor de Matterhorn, para poder visualizar la grabación.

11. En este momento ya es posible visualizar la grabación realizada en el reproductor de Matterhorn, teniendo disponibles todas las opciones mencionadas en la sección dedicada al Reproductor.

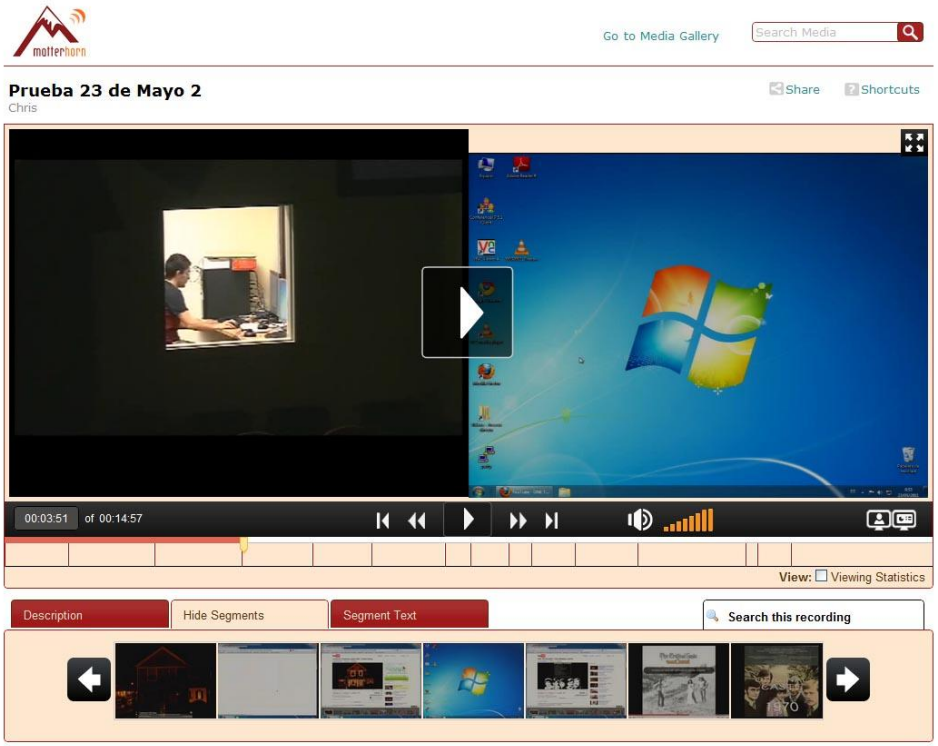


Ilustración 26 Ejemplo Grabación 11. Reproducción de la grabación realizada en el reproductor

4. Pasarela ArcaMM

Cuando se han terminado las operaciones de procesado de un vídeo, éste ya está disponible para su visualización en el reproductor de Matterhorn, pero no con el comportamiento deseable ya que Matterhorn presenta limitaciones a la hora de organizar de los vídeos generados.

Al disponer de la plataforma ArcaMM donde se encuentra toda la información relativa al material audiovisual generado por la universidad, lo ideal es que la información y el acceso a los vídeos se realicen a través de dicha plataforma

Es necesario por tanto crear un sistema que permita agregar a ArcaMM los datos referentes a los vídeos grabados por Matterhorn para su posterior catalogación y puesta a disposición de la comunidad universitaria a través de la mencionada plataforma.

4.1. Análisis de situación.

Una vez se encuentre en funcionamiento el sistema Matterhorn, es necesario observar qué datos se pueden extraer de las grabaciones realizadas.

Para ello en las **Tabla 8 y Tabla** 9, se observan los datos de los feeds del reproductor, que se encuentran disponibles.

Feed RSS:

```
<item>
      <title>Máster Interuniversitario en Multimedia y Comunicaciones
2</title>
      <link>http://163.117.69.143:8080/engage/ui/watch.html?ID=1454</link>
      <enclosure    url="http://163.117.69.143:8080/static/1454/6ba17262-794b-
4106-ac0c-f433f0c997fc/VGA.avi" length="-1" type="vídeo/avi" />
      <pubDate>Tue, 05 Jul 2011 13:55:00 GMT</pubDate>
      <guid isPermaLink="false">1454</guid>
      <itunes:duration>01:57:31</itunes:duration>
      <itunes:author>Mª Eugenia Abad</itunes:author>
      <itunes:explicit>no</itunes:explicit>
      <itunes:keywords />
      <dc:title>Máster Interuniversitario en Multimedia y Comunicaciones
2</dc:title>
      <dc:creator>Mª Eugenia Abad</dc:creator>
      <dc:date>2011-07-05T13:55:00Z</dc:date>
      <dc:identifier>1454</dc:identifier>
      <dc:language>EN</dc:language>
    </item>
```

Tabla 8 Ejemplo Feed RSS ofrecido por Matterhorn

Feed ATOM:

```
  <entry>
    <title   mode="escaped">Máster   Interuniversitario   en   Multimedia   y
Comunicaciones 2</title>
    <link href="http://163.117.69.143:8080/engage/ui/watch.html?ID=1454" />
    <link                  rel="enclosure"                  type="vídeo/avi"
href="http://163.117.69.143:8080/static/1454/6ba17262-794b-4106-ac0c-
f433f0c997fc/VGA.avi" />
    <author>
      <name>Mª Eugenia Abad</name>
    </author>
    <ID>1454</ID>
    <modified>2011-07-05T13:55:00Z</modified>
    <issued>2011-07-05T13:55:00Z</issued>
    <dc:title>Máster Interuniversitario en Multimedia y Comunicaciones
2</dc:title>
    <dc:creator>Mª Eugenia Abad</dc:creator>
    <dc:date>2011-07-05T13:55:00Z</dc:date>
    <dc:identifier>1454</dc:identifier>
    <dc:language>EN</dc:language>
  </entry>
```

Tabla 9 Ejemplo Feed ATOM ofrecido por Matterhorn

Como se puede observar los distintos datos obtenidos, son los siguientes:

- Datos formato Estándar (ver **Tabla 10**)
- Datos formato iTunes RSS (ver **Tabla 11**)
- Datos del metamodelo de datos Dublin Core (ver **Tabla 12**)

• Título de la grabación • Enlace para la visualización de la grabación

• Fecha publicación • ID de la grabación • Autor (autores si los hay). • Enclosure, que consta de tres atributos, URL del archivo para descarga, tipo de archivo, duración

Tabla 10 Datos formato estándar obtenidos del feed proporcionado por Matterhorn

• Duración H:M:S • Autor • Información sobre si el vídeo requiere información parental. • Palabras claves usadas para la búsqueda

Tabla 11 Datos iTunesRSS obtenidos del feed proporcionado por Matterhorn

• Titulo de la grabación • Autor de la grabación • Fecha de creación • ID del vídeo. • Idioma de grabación.

Tabla 12 Datos Dublin Core obtenidos del feed proporcionado por Matterhorn

Mediante el RSS se facilitan en total 9 datos que se podrán utilizar para su posterior integración en ArcaMM.

La plataforma ArcaMM incorpora un sistema para añadir vídeos a su catálogo. Este sistema requiere de diversos datos para poder catalogar un ítem, que comprenden, aparte de los anteriormente mencionados, datos como calidad del vídeo, categoría a la que pertenece el vídeo, datos relativos acerca de la privacidad del vídeo, etc.

El objetivo para la siguiente fase del proyecto es la transferencia de datos de Matterhorn a ArcaMM.

La primera operación es intentar utilizar el feed que proporciona Matterhorn, pero esta operación resulta insuficiente debido a la información plasmada en dicho feed.

Se adopta entonces la decisión de diseñar para el sistema ArcaMM un WebService que permita introducir la mayor cantidad de datos posible en la aplicación de catalogación, dejando al catalogador introducir aquellos datos que falten por completar.

Es necesario diseñar un sistema que a partir de un dato, sea capaz de generar toda la información posible para su posterior procesado en ArcaMM (a través del WebService mencionado anteriormente, aspecto que queda fuera del alcance de este proyecto).

Ilustración 27 Esquema Inicial Pasarela.

4.2. Análisis de datos necesarios.

Con el fin de facilitar el proceso de insertar datos de los vídeos en la base de datos de ArcaMM en necesario conocer primero que datos son necesarios.

Por un lado se dispone de los datos descriptivos de la grabación, como el título, el autor, fecha, ponente, etc.

Otros son relativos a los datos técnicos del vídeo, como formato, tamaño, duración, bitrate, framerate, carátulas, etc.

Por último existen otros tipos de datos como puede ser categoría del vídeo, serie a la que pertenecen, que se introducirán de forma natural cuando se catalogue el vídeo y por tanto no son necesarios en esta fase.

En cuanto al primer tipo de datos, son facilitados por el feed RSS, por lo que con proceder a su extracción ya estarían disponibles.

De estos datos, se ha proyectado como necesarios los siguientes:

- Titulo Grabación.

- Fecha Grabación.
- ID Grabación.
- Autor(es).
- Idioma grabación.
- URLVisor.
- Subtítulos (existen o no).

Mencionar que de todos los datos anteriores, el único que no se puede obtener directamente es relativo a la existencia o no de los subtítulos. Pero ese dato se extraerá en una fase posterior.

Se necesitan también datos técnicos de los vídeos grabados, que sólo se pueden obtener analizando directamente los ficheros. Para ello, es necesario diseñar e implementar un sistema que permita este análisis.

Por el propio diseño de Matterhorn, en principio existen dos vídeos de los que han de analizarse, el vídeo de la grabación del presentador, y el vídeo de la grabación de la señal VGA.

Existe también un tercer vídeo, de generación opcional, aunque útil para la integración con ArcaMM. Es el vídeo de la señal VGA disponible para descarga (que como se ha mencionado anteriormente, no tiene el mismo formato que la versión destinada al reproductor).

También es necesario disponer de una imagen JPG de la grabación del Presentador, imagen que se puede elegir de entre las generadas por Matterhorn, pero que requiere un análisis previo como los anteriores elementos.

4.3. Diseño de Base de Datos Utilizado.

En base a los datos especificados en la sección anterior se ha propuesto el siguiente diseño para la base de datos que se va a utilizar.

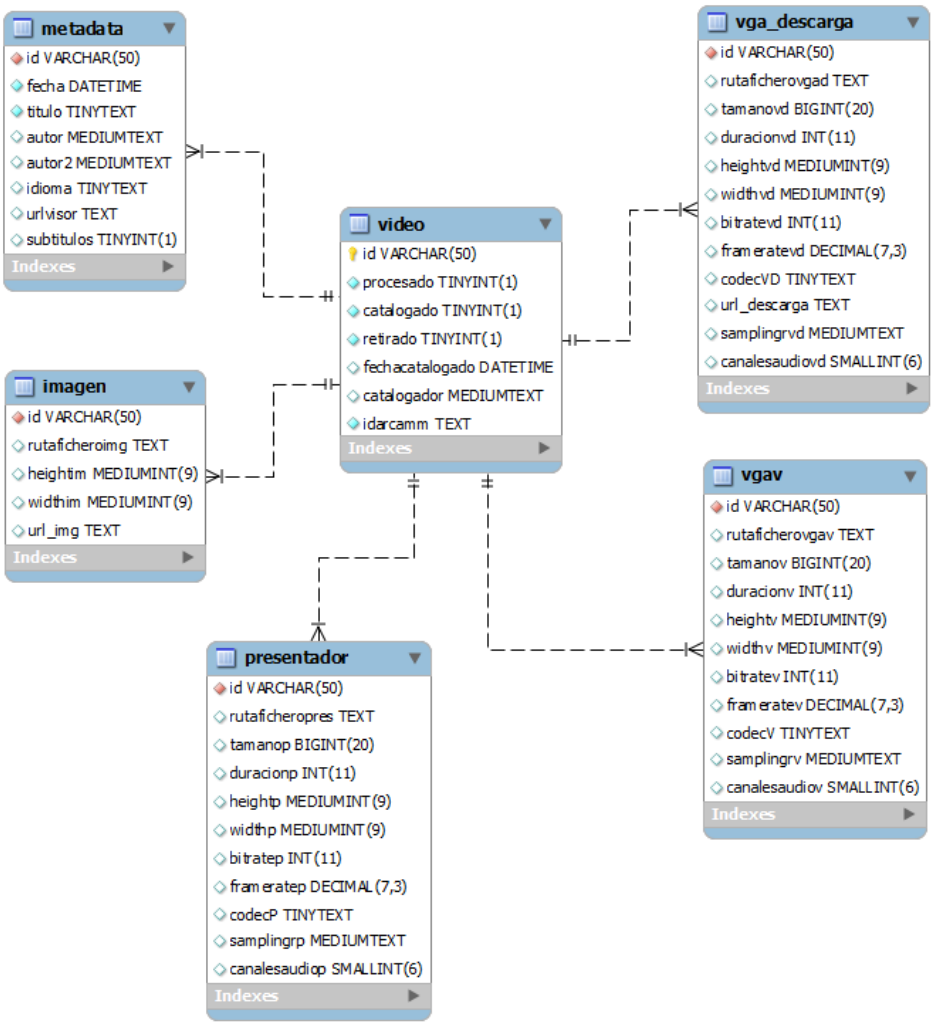


Ilustración 28 Diagrama Entidad/Relación de Base de datos de la aplicación.

En este diseño de base de datos, aparece una tabla central, llamada VIDEO, que es la tabla principal, y que contiene datos susceptibles de modificación, datos como si el vídeo ha sido procesado, catalogado o retirado de la plataforma ArcaMM. También figura el identificador de este vídeo, en caso de que haya sido catalogado en ArcaMM.

Esta tabla incorpora un campo, el campo “Catalogador”, que indicará, una vez que el vídeo se haya incorporado a la base de datos de ArcaMM, el nombre de la persona que ha validado ese vídeo.

Este campo lo obtendrá como respuesta del Web Service que añadirá los datos a la plataforma ArcaMM, junto con el anterior campo mencionado de la ID de ArcaMM.

En tabla “VIDEO”, el campo principal es “ID”, que es una clave primaria de la tabla y coincide con el identificador propio de cada vídeo. Este campo nunca puede ser nulo.

El resto de tablas contienen datos relativos a los elementos que se han analizado previamente y que son de interés para el proceso, datos relativos a los ficheros de vídeo, a la imagen y metadatos propios de cada grabación.

Estas tablas están relacionadas con la tabla VÍDEO a través de la clave ID, que en estas tablas es clave foránea desde la tabla VÍDEO.

Además esta clave es de tipo UNIQUE, con lo que se evita tener que controlar manualmente que no se produzcan duplicados de las entradas. Con este método, directamente MySQL impedirá que eso ocurra.

Los tipos de datos para los campos de cada tabla se han escogido en función de los datos que se han de almacenar y para aquellos casos en los que únicamente se requería un booleano, MySQL lo crea de forma interna como un dato TINYINT, ya que para MySQL ambos sinónimos.

4.4. Extractor de datos.

Para poder catalogar posteriormente los vídeos que se han grabado con Matterhorn en ArcaMM, es necesario disponer de diversos datos, relativos tanto a los mismos vídeos (título, autor, fecha, etc.), como datos técnicos de los elementos grabados (vídeos del presentador y de la señal VGA, datos de la imagen, etc.).

El primer tipo de datos, como se ha mencionado anteriormente, es sencillo de obtener, ya que esa información es facilitada por los feeds RSS de los que dispone Matterhorn.

El problema se genera con los datos técnicos, necesarios para la correcta catalogación en ArcaMM, ya que estos no aparecen de forma automática en ningún feed de los proporcionados y por tanto es necesario proceder a su búsqueda.

Para ello, se buscará en el propio sistema de archivos donde se encuentran almacenadas los vídeos de las grabaciones para acceder a ellos.

En una instalación típica de Matterhorn, todos los elementos multimedia destinados a la reproducción en el visor, se encuentran dentro de la carpeta *downloads*.

Esta carpeta está dividida a su vez en diversas subcarpetas que contienen todos los archivos de cada una de las grabaciones.

Estas carpetas están identificadas con la ID de la grabación realizada. Cuando una grabación tiene como URL similar a la siguiente:

```
http://163.117.69.143:8080/engage/ui/watch.html?ID=72
```

Todos los archivos asociados a la grabación se encontraran en la carpeta `<directorio de trabajo de Matterhorn>/downloads/72`.

De esta forma, se podrá realizar una búsqueda dentro de este directorio para localizar los archivos multimedia que se necesita analizar para obtener sus metadatos.

Esta es en líneas generales la función del “Extractor”, el script programado con el lenguaje PHP.

Para que disponer de dicha funcionalidad, se decide que el script realice para cada grabación 5 extracciones de datos distintas:

1. Extracción de Metadatos
2. Extracción de Datos de Imagen para caratula
3. Extracción de datos del vídeo del presentador
4. Extracción de datos del vídeo de la señal VGA

5. Extracción de datos del vídeo de la señal VGA apta para descarga.

Estos datos se irán almacenando en un array con 41 campos, cada uno correspondiente a alguno de los datos que se van a necesitar (Ver **Tabla 13**).

```
$datos= Array();
$datos['ID'];
$datos['fecha'];
$datos['titulo'];
$datos['autor'];
$datos['autor2'];
$datos['link'];
$datos['idioma'];
$datos['subtitulos'];
$datos['rutafimagen';
$datos['imgheight'];
$datos['imgwidth'];
$datos['urlimagen'];/*url imagen*/
$datos['rutafichvídeoP'];
$datos['tamanop'];
$datos['vídeoheightP'];
$datos['vídeowidthP'];
$datos['vídeobitrateP'];
$datos['vídeoframerateP'];
$datos['codecP'];
$datos['samplingrp'];
$datos['canalaudiop'];
$datos['rutafichvídeoV'];
$datos['tamanov'];
$datos['vídeoheightV'];
$datos['vídeowidthV'];
$datos['vídeobitrateV'];
$datos['vídeoframerateV'];
$datos['codecV'];
$datos['samplingrv'];
$datos['canalaudiov'];
$datos['rutafichvídeoVD'];
$datos['tamanovd'];
$datos['vídeoheightVD'];
$datos['vídeowidthVD'];
$datos['vídeobitrateVD'];
$datos['vídeoframerateVD'];
$datos['codecVD'];
$datos['samplingrv'];
$datos['canalaudiov'];
$datos['urlvgadesc'];
$datos['duracion'];
```

Tabla 13 Array para datos del vídeo

En la medida que se vayan extrayendo datos, se irán almacenando en su campo correspondiente de este array.

- **Funcionamiento Script.**

El funcionamiento básico del script es el siguiente (posteriormente se explicará cada apartado en profundidad)

`<Inicializacion Parseador>`	Se inicializa el funcionamiento del parseador SimplePie (Ver **Tabla 15**)
`<Activar conexion con base de datos>`	Se abre la conexión con la base de datos que se utilizará.
`<Definicion de Array de datos>`	Se define el array de datos donde se almacenarán los datos que se vayan extrayendo (Ver **Tabla 13**)
`$dirmedios="http://163.117.69.143:8080/static/";`	Variable que indica la URL accesible de los elementos que componen la grabación
`foreach($feed->get_items(0,0) as $item):`	Bucle que se ejecuta mientras se encuentren en el feed el número de elementos especificados, definiendo en cada pasada del bucle un objeto *ítem* que contiene los elementos de esa entrada del feed
`$ID=$item->get_ID();`	Obtiene del ítem seleccionado su ID de grabación, utilizada para extraer los datos.
`$directorio = '/mnt/matterhorn/opencast/work/downloads/'.$ID;`	Junto con la ID anteriormente obtenido, define el directorio en el que se encuentran los ficheros de la grabación a analizar

`$muestra ='SELECT * FROM vídeo where ID="'.$ID.'" AND procesado=true';`
`$muestracon = mysql_query($muestra);`
`$muestrares = mysql_num_rows($muestracon);`
	Realiza una consulta a la tabla **VÍDEO** de la base de datos para comprobar si la grabación se encuentra en la BD.

`if ($muestrares>0){` `}else{`	En caso de que la grabación no se encuentre procesada, se comienza con el análisis de elementos y su procesamiento. Si está procesada continua con el siguiente elemento.
`<extractor metadatos>`	Operación que extrae metadatos del vídeo (Ver **Sección Propia**)

`$subex=exec('find '.$directorio.' -name *.dfxp');`
`if ($subex==NULL){`
`$datos[subtitulos]='0';`
`}else{`
`$datos[subtitulos]='1';`
`}`
	Operación que mediante la función *exec*() de PHP permite realizar operaciones externas en el SO, en este caso una búsqueda en caso de que exista fichero de subtítulos, tras lo que marcara en el array su existencia

`<extractor datos imagen>`	Operación que extrae metadatos del vídeo (Ver **Sección Propia**)

`<extractor datos vídeo presentador>`	Operación que extrae metadatos del vídeo (Ver **Sección Propia**)
`<extractor datos vídeo VGA>`	Operación que extrae metadatos del vídeo (Ver **Sección Propia**)
`<extractor datos vídeo VGA descarga>`	Operación que extrae metadatos del vídeo (Ver **Sección Propia**)

`$muestra2="SELECT * FROM vídeo WHERE ID='$ID'";`
`$muestra2=mysql_query($muestra2);`
`$muestrares2 = mysql_num_rows($muestra2);`
	Realiza una consulta a la tabla **VÍDEO** de la base de datos para comprobar si la grabación se encuentra en la BD.

`if ($muestrares2>0){`
`}else{`
`<insercion en base de datos>`
`}`
	Si la grabación no está procesada, se insertan los datos del array *Datos* en la base de datos

`unset($datos);` `}`	Vacía el array de datos para que no se produzcan valores erróneos en la siguiente iteración del script.
`endforeach;`	Finaliza el bucle de análisis de elementos cuando se han inspeccionado todos los elementos del feed
`mysql_close($conexion);`	Se cierra la conexión con la base de datos

Tabla 14 Funcionamiento Script Extracción

- **Extracción de Metadatos.**

La información de los metadatos necesarios se puede obtener tras tratar el feed RSS (se exceptúa la información de la existencia o no de subtítulos).

Para ello se utilizará la librería SimplePie, que permite parsear los feeds RSS y ATOM de una forma rapida. Así se obtendrán los metadatos del vídeo a analizar que ha proporcionado el feed.

La primera operación consiste en inicializar en la cabecera del fichero PHP el script parseador SimplePie para extraer datos del feed.

```
include_once("/var/www/app/simplepie.inc");
$feed = new SimplePie();

$feed->set_feed_url("http://163.117.69.143:8080/feeds/rss/2.0/latest");
$feed->enable_cache(false);
$feed->enable_xml_dump(isset($_GET['xmldump']) ? true : false);

$success = $feed->init();
```

Tabla 15 Inicialización Parseador SimplePie

El funcionamiento del código mostrado en la **Tabla 15** es el siguiente:

1. En primer lugar se incluye la ruta donde se encuentra el archivo de SimplePie.
2. Una vez cargado ese archivo, se crea una instancia del objeto SimplePie
3. Posteriormente se configuran los siguientes parámetros:
    - Feed que se quiere buscar, en este caso el feed RSS que proporciona Matterhorn: http://163.117.69.143:8080/feeds/rss/2.0/latest
    - Posteriormente se indica a SimplePie que no se quiere que guarde una cache del feed analizar. Se desea que siempre utilice el feed de nuevo, para evitar tener elementos desactualizados
    - Finalmente se configura la salida del resultado de parsear el feed a formato XML, para poder tratarla con más facilidad.
4. Para finalizar, se inicializa el feed, con lo que ya se puede manejar como si de un objeto se tratara

Una vez realizadas estas operaciones y tras inicializar el array de datos, se inicializa la búsqueda de elementos con el siguiente bucle *foreach* :

```
foreach($feed->get_items(0,0) as $item):
```

De esta forma en cada iteración del bucle hay un elemento *$item* que proviene del feed y que será con el se trabaje en cada pasada del bucle.

Tras obtener mediante la siguiente sentencia: `$ID=$item->get_ID();` la ID del elemento y que el sistema tome la decisión requirida, en base a los datos anteriormente almacenados, se determinara si se procesa el vídeo o si ya se habían añadido los datos a la BD y por tanto no es necesario realizar el análisis.

Por último se procede a extraer el resto de metadatos del feed y a introducirlos en el array de datos.

```
$datos[ID]= $ID;
$datos[titulo]= $item->get_title();
$datos[link]= $item->get_permalink();
$datos[fecha]= $item->get_date('Y-m-d H:i:s');
```

Con estas funciones de la librería SimplePie se obtienen datos del feed y se introducen en el array de datos.

Estas funciones lo que hacen es obtener un dato concreto de una etiqueta específica, como por ejemplo la función *get_title()* que obtiene el valor que hay en la etiqueta **Title**: `<title>Grabación del dia X</title>`. En este caso, se obtiene el valor "Grabación del dia X" y se introduce en su variable correspondiente.

La función *get_date()* presenta unas características especiales, ya que si bien acepta el valor de la etiqueta **pubDate**, permite modificar el formato de esta fecha, pudiendo elegir el formato de salida, en este caso al formato de hora por defecto de MySQL.

Los parámetros que acepta para la modificación del formato de hora son los mismos que acepta la función *Date()*, función nativa de PHP.

```
$autor = $item-> get_item_tags(SIMPLEPIE_NAMESPACE_DC_11,'creator');
$datos[autor]= $autor[0][data];
$autor2 = $item-> get_item_tags(SIMPLEPIE_NAMESPACE_DC_11,'contributor');
$datos[autor2]= $autor2[0][data];
$idioma = $item-> get_item_tags(SIMPLEPIE_NAMESPACE_DC_11,'language');
$datos[idioma]= $idioma[0][data];
```

Obtener los datos que se muestran utilizando para ello el modelo de metadatos Dublin Core requiere un proceso de trabajo no tan directo como los anteriores, pero sencillo de implementar.

SimplePie permite obtener datos de un variado número de *namespaces*, entre los que se encuentran los namespaces de Dublin Core, iTunesU.

La función get_item_tags(), requiere dos parámetros: el sistema de *namespace* que se quiere utilizar y la etiqueta en concreto que se quiere seleccionar.

El resultado es un array bidimensional presenta la siguiente estructura

```
Array(
       [0] => Array(
              [data]=> 'Valor correspondiemte'
             )
)
```

Tabla 16 Ejemplo de Array Devuelto por get_item_tags()

Como se observa, los valores obtenidos desde el RSS no se pueden utilizar directamente, por lo que la solución utilizada es crear variables en las que almacenar

este array que devuelven las funciones *get_item_tags()* y posteriormente utilizar estas variables para extraer el valor deseado y almacenarlo en el array de datos.

```
$autor = $item-> get_item_tags(SIMPLEPIE_NAMESPACE_DC_11,'creator');
$datos[autor]= $autor[0][data];
```

Tabla 17 Introducción de datos del autor en el array

Como se puede apreciar en el ejemplo de la **Tabla 17**, la función get_item_tags() requiere de los dos parámetros anteriormente citados, que en este caso son **`SIMPLEPIE_NAMESPACE_DC_11`** y **`'creator'`**.

`SIMPLEPIE_NAMESPACE_DC_11` es el parámetro que indica que el dato que se quiere extraer pertenece al metamodelo Dublin Core versión 1.1.

`'Creator'` indica que se quiere obtener el contenido de la etiqueta **`dc:Creator`** (las etiquetas de los datos del metamodelo Dublin Core van precedidas de **`"dc:"`**).

Con esta operativa se obtiene el dato y que como anteriormente se ha comentado, resulta ser un array del que posteriormente se extrae el dato concreto que se necesita y se almacena en su variable correspondiente del array de datos.

- **Extracción de datos de la imagen.**

El funcionamiento esencial del extractor consiste en el uso de la función exec() de PHP.

Esta función permite ejecutar un comando del sistema operativo y que el resultado pueda ser manipulado desde PHP.

La forma estándar de uso de esta función es `exec(<comando>)` .

Por ejemplo la siguiente función:

```
<?php
echo exec('whoami');
?>
```

Ejecutada sobre un servidor en el que exista la función *whoami*, devolverá el nombre del usuario que está ejecutando el archivo sobre el servidor.

Este retorno puede ser manipulado como el programador decida, ya sea mostrándolo, utilizando para realizar cálculos, o simplemente almacenándolo, siendo este último el uso principal que se le dé en este script.

```
$rutafichf=exec('find '.$directorio.' -name Presentador.jpg');
$datos[rutafimagen]=$rutafichf;
$datos[imgheight]=exec('mediainfo "--Inform=Image;%Height%" '.$rutafichf);
$datos[imgwidth]=exec('mediainfo "--Inform=Image;%Width%" '.$rutafichf);
$datos[urlimagen]= $dirmedios.substr($rutafichf,40);
```

Tabla 18 Código del extractor de datos de la imagen

Para poder obtener los datos necesarios de la imagen que servirá de caratula para ArcaMM.

En primer lugar es necesario localizar la imagen en el disco duro. Matterhorn siempre genera a la hora de crear miniaturas para sus feed, como mínimo una imagen llamada *Presentador.jpg* y otra imagen *VGA.jpg*.

El Área de Audiovisuales decidió que la imagen que interesaba al proyecto es una captura de la ponencia, o por lo menos, del lugar donde el ponente habla, por lo que de aquí en adelante se descartarán todas las imágenes *VGA.jpg.*

Para ello se utiliza la función *exec*() cuyo valor se almacenará para después poder utilizar esa ruta en el analizador.

El único parámetro necesario es el siguiente:

```
exec('find '.$directorio.' -name Presentador.jpg')
```

Tabla 19 Parámetro Exec() para la búsqueda de la imagen

El objeto de esta función con este parámetro es realizar una búsqueda en el directorio donde se encuentran los datos del vídeo que se analiza (la variable `$directorio = '/mnt/matterhorn/opencast/work/downloads/<ID del vídeo>'`) del archivo con nombre Presentador.jpg, tras lo que devuelve la ruta en la que se encuentra dicho fichero.

A continuación, se utiliza esa ruta para realizar la llamada al analizador **MediaInfo**:

```
$datos[imgheight]=exec('mediainfo "--Inform=Image;%Height%" '.$rutafichf);
```

```
$datos[imgwidth]=exec('mediainfo "--Inform=Image;%Width%" '.$rutafichf);
```

En esta situación lo que se ejecuta tras lanzar la función *exec*(), es el programa M**ediaInfo**, con los parámetros que indican que se quiere obtener la altura y la anchura, respectivamente, de la imagen, cuya ruta es donde se encuentra almacenada con anterioridad en la variable *$rutafichf* .

Como último paso para disponer de todos los ficheros relativos a la imagen, queda obtener su URL pública.

Teniendo en cuenta que parte de las rutas de los ficheros en el sistema de archivos coincide con parte de la IP pública de los mismos, para obtener la URL del fichero en cuestión es necesario eliminar una parte de la ruta del fichero, y sustituir esa parte por la URL desde la cual se podrá acceder a la imagen seleccionada desde el exterior.

Dicho proceso se realiza mediante la siguiente operación:

```
$datos[urlimagen]= $dirmedios.substr($rutafichf,40);
```

que elimina los caracteres que corresponden a la localización del fichero en el sistema de archivos (en la instalación actual son 40 caracteres) y concatena a la ruta restante el valor de la variable $*dirmedios*, en este caso `http://163.117.69.143:8080/static/`.

De esta forma disponemos de un enlace directo a una imagen en concreto, así como a sus datos de altura y anchura.

- ## Extracción de datos del vídeo del Presentador, VGA y VGA para Descarga.

Para extraer el resto de datos de los restantes elementos multimedia, el proceso a seguir es muy similar al realizado con anterioridad para extraer metadatos de la imagen.

```
$fichbusc ='Presentador.f4v';
$fichbusc2 ='Presentador.flv';
$rutafichPV=exec('find '.$directorio.' -name '.$fichbusc);
if ($rutafichPV==NULL){
       $rutafichPV=exec('find '.$directorio.' -name '.$fichbusc2);
}
$datos[rutafichvídeoP]= $rutafichPV;
$datos[vídeobitrateP]=exec('mediainfo "--Inform=Vídeo;%BitRate%"
'.$rutafichPV);
$datos[vídeoheightP]=exec('mediainfo "--Inform=Vídeo;%Height%" '.$rutafichPV);
$datos[vídeowidthP]=exec('mediainfo "--Inform=Vídeo;%Width%" '.$rutafichPV);
```

```
$datos[duracion]=exec('mediainfo "--Inform=Vídeo;%Duration%"
'.$rutafichPV)/1000;
$datos[vídeoframerateP]=exec('mediainfo "--Inform=Vídeo;%FrameRate%"
'.$rutafichPV);
$datos[codecP]=exec('mediainfo "--Inform=Vídeo;%Format%" '.$rutafichPV);
$datos[samplingrp]=exec('mediainfo "--Inform=Audio;%SamplingRate/String%"
'.$rutafichPV);
$datos[canalaudiop]=exec('mediainfo "--Inform=Audio;%Channel(s)%"
'.$rutafichPV);
$datos[tamanop]=exec('mediainfo "--Inform=General;%FileSize%" '.$rutafichPV);
```

Tabla 20 Código extractor datos vídeo presentador.

En el caso contemplado en la **Tabla 20** se ha introducido una comprobación adicional, relativa al nombre del fichero.

Actualmente Matterhorn no utiliza códecs H.264 para la reproducción en el *Engage*, debido a que el reproductor no sirve correctamente ficheros con esta codificación. Esta funcionalidad deberá ser mejorada en futuras versiones del proyecto.

Actualmente los ficheros no utilizan el códec H.264, por lo que no utilizan el contenedor vinculado para esta codificación (contenedor F4V). En su lugar se usan contenedores FLV.

La comprobación introducida consiste en tener definidos los posibles nombres de fichero que pueda tener la grabación del vídeo del presentador.

En el escenario actual, los posibles contenedores a utilizar (y por tanto, extensiones del archivo), son contenedores FLV y F4V.

De esta forma, y como se ha explicado anteriormente al realizar la búsqueda de la imagen, se procede a encontrar el fichero de vídeo que se necesitará analizar.

```
$fichbusc ='Presentador.f4v';
$fichbusc2 ='Presentador.flv';
$rutafichPV=exec('find '.$directorio.' -name '.$fichbusc);
if ($rutafichPV==NULL){
      $rutafichPV=exec('find '.$directorio.' -name '.$fichbusc2);
}
```

Tabla 21 Buscador de ruta de fichero de vídeo

En primer lugar, la búsqueda se realiza utilizando uno de los dos nombres de archivo previamente definidos.

Únicamente en el caso de que el fichero buscado no se encuentre, porque se ha utilizado la otra extensión, se realiza otra búsqueda con el otro nombre del fichero definido, para poder obtener la ruta y proceder a su posterior análisis.

Una vez conocida la ruta se procede a lanzar el analizador **MediaInfo** que permite extraer los datos de altura y anchura del vídeo, duración, bitrate, códec utilizado, canales de audio disponibles en el vídeo, frecuencia de muestreo del audio, así como el tamaño del fichero. Esta información se almacenará en el array de datos.

Este proceso se realiza para los 3 tipos de vídeo, Presentador, VGA y VGA para descarga, diferenciándose únicamente en el nombre del fichero a buscar. El resto del proceso, por tanto, es el mismo: localización de ruta y extracción de datos.

- ## Extracción de datos extra para el vídeo VGA para Descarga.

Para la extracción de datos del vídeo VGA para descarga, es necesario añadir un dato adicional, la URL pública del archivo. Esta URL no se obtiene el mismo proceso que se utilizaba para obtener la imagen, seccionando la ruta del fichero obtenida y añadiéndole la parte de la URL, sino que, en este caso, el enlace se obtiene desde el feed RSS, mediante la extracción del contenido de la etiqueta *enclosure*, utilizando la librería **SimplePie:**

```
if ($enclosure = $item->get_enclosure()){
        $datos[urlvgadesc]= $enclosure->get_link();
}
```

Tabla 22 Obtención del link desde el enclosure.

Realizada esta acción, se obtendrá el dato que faltaba y se añadirá al array de datos.

- ## Inserción del array de datos en la base de datos.

Una vez que se han obtenido todos los datos posibles de la grabación, llega el momento de introducirlos en la base de datos disponible.

A la hora de realizar consultas, y para evitar posibles errores en el uso de transacciones largas, se adopta la decisión de dividirlas en 5, una por cada tabla de la base de datos.

Por tanto, antes de empezar a introducir datos, es necesario crear una tabla, la tabla VIDEO, que será la tabla principal, con una nueva entrada que incluye la ID del vídeo cuyos datos se van a almacenar.

Es necesario que en esta tabla exista una fila con la ID del vídeo que se está utilizando ya que de lo contrario, el resto de tablas no permitirán que se introduzcan datos de ese vídeo en concreto, puesto que el campo ID del resto de tablas son claves foráneas, que dependen de la existencia de este campo en la tabla VÍDEO.

A continuación, se introducen en cada tabla los datos correspondientes, obtenidos desde el array de datos.

```
$consulta="INSERT INTO vídeo (ID) VALUES ('$datos[ID]')";
$insercion=mysql_query($consulta);

$consulta="INSERT INTO metadata
(ID,fecha,titulo,autor,autor2,urlvisor,idioma,subtitulos) VALUES
('$datos[ID]','$datos[fecha]','$datos[titulo]','$datos[autor]','$datos[autor2]
','$datos[link]','$datos[idioma]','$datos[subtitulos]')";
$insercion=mysql_query($consulta);

$consulta="INSERT INTO imagen (ID,rutaficheroimg,heightim,widthim,url_img)
VALUES
('$datos[ID]','$datos[rutafimagen]','$datos[imgheight]','$datos[imgwidth]','$d
atos[urlimagen]')";
$insercion=mysql_query($consulta);

$consulta="INSERT INTO presentador
(ID,rutaficheropres,heightp,widthp,duracionp,bitratep,frameratep,codecP,sampli
ngrp,canalesaudiop,tamanop) VALUES
('$datos[ID]','$datos[rutafichvídeoP]','$datos[vídeoheightP]','$datos[vídeowid
thP]','$datos[duracion]','$datos[vídeobitrateP]','$datos[vídeoframerateP]','$d
atos[codecP]','$datos[samplingrp]','$datos[canalaudiop]','$datos[tamanop]')";$
insercion=mysql_query($consulta);

$consulta="INSERT INTO vgav
(ID,rutaficherovgav,heightv,widthv,duracionv,bitratev,frameratev,codecV,sampli
ngrv,canalesaudiov,tamanov) VALUES
('$datos[ID]','$datos[rutafichvídeoV]','$datos[vídeoheightV]','$datos[vídeowid
thV]','$datos[duracion]','$datos[vídeobitrateV]','$datos[vídeoframerateV]','$d
atos[codecV]','$datos[samplingrv]','$datos[canalaudiov]','$datos[tamanov]')";
$insercion=mysql_query($consulta);

$consulta="INSERT INTO vga_descarga
(ID,url_descarga,rutaficherovgad,heightvd,widthvd,duracionvd,bitratevd,framera
tevd,codecVD,samplingrvd,canalesaudiovd,tamanovd) VALUES
('$datos[ID]','$datos[urlvgadesc]','$datos[rutafichvídeoVD]','$datos[vídeoheig
htVD]','$datos[vídeowidthVD]','$datos[duracion]','$datos[vídeobitrateVD]','$da
tos[vídeoframerateVD]','$datos[codecVD]','$datos[samplingrvd]','$datos[canalau
diovd]','$datos[tamanovd]')";
$insercion=mysql_query($consulta);

$consulta="UPDATE vídeo set procesado=1";
$insercion=mysql_query($consulta);
```

Tabla 23 Inserción datos de vídeo en la base de datos

Para finalizar, una vez procesada la grabación, se actualiza la tabla VIDEO marcando la misma como procesado. (Valor procesado = 1)

Una vez introducidos los datos del vídeo en la base de datos, se procede a vaciar el array de datos, con el fin de dejarlo preparado para el siguiente vídeo a analizar.

```
unset($datos);
```

Cuando se hayan procesado todos los vídeos, se debe cerrar la conexión con la base de datos, finalizando la ejecución del script.

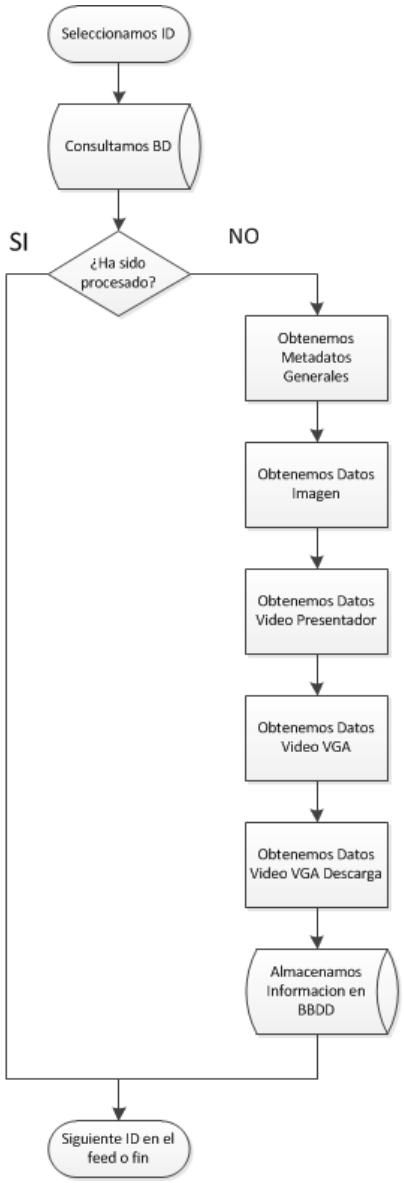


Ilustración 29 Diagrama Flujo Extractor de Datos

- **Ejecución del script.**

El último paso necesario para que el extractor funciones es su ejecución. Para evitar ejecutarlo manualmente, se programa la ejecución del script mediante el programa **Cron**, automatizando el proceso de inserción de nuevas grabaciones de Matterhorn en nuestra base de datos.

Para programar esta operación, se edita nuestro fichero crontab con el comando `crontab –e`.

Realizada la acción anterior, aparece una pantalla donde se podrá programar la ejecución del script. En esa pantalla se añadirá la siguiente línea según figura en la **Tabla 24**.

```
* 4 * * *  /usr/bin/php /<ruta donde se encuentre el extractor>/extractor.php
```

Tabla 24 Ejemplo de Cron para la ejecución del extractor

Esta línea deja programada todos los días a las 4 de la mañana, la ejecución del comando "`/usr/bin/php /<ruta>/extractor.php`", que lanzará el script a través de PHP.

De esta forma se consigue que todas las noches y de forma automática, la base de datos quede actualizada con las nuevas grabaciones realizadas el día anterior.

4.5. Integrador con ArcaMM.

Una vez almacenados los datos en la base de datos, es necesario diseñar una interfaz que permita manipular e incorporar a la base de datos de ArcaMM los metadatos de los vídeos grabados.

En primer lugar es necesario diseñar e implementar una página web a través de la cual poder visualizar y seleccionar los vídeos grabados que se quieran incorporar a ArcaMM, así como retirar vídeos cuando no se requiera que estén disponibles para su visualización en ArcaMM.

- **Visión General.**

Esta web formará parte, como una sección más dentro de ArcaMM, por lo que al desarrollar la interfaz web, se debe procurar que esté integrada, en la medida de lo posible, con el resto de la plataforma. Para ello se reutilizarán las hojas de estilo disponibles en la plataforma, así como los scripts en uso.

Para mostrar los vídeos disponibles, se usarán los datos almacenados anteriormente en la base de datos.

Mediante esa base de datos, se obtendrán los datos de los vídeos, y los podrá mostrar por cada una de las 3 categorías disponibles (*Pendientes*, *Catalogados* y *Retirados)* en función de si ya se ha gestionado o no esa grabación. En la página de gestión de grabaciones, estarán disponibles las 3 categorías anteriormente mencionadas.

El aspecto y funcionamiento básico es esencialmente el mismo en las 3 categorías, por lo que se va a describir en primer lugar el funcionamiento básico y a continuación, las particularidades concretas de cada una de ellas.

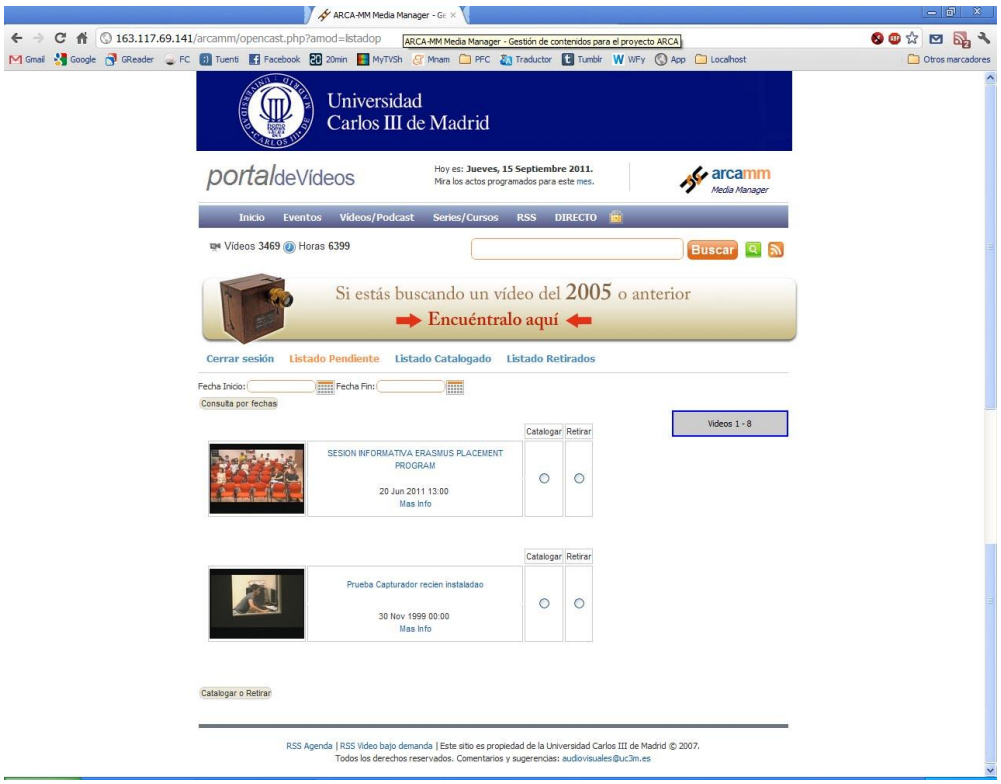


Ilustración 30 Aspecto Interfaz Web Opencast-ArcaMM

En la interfaz web se puede observar, que existen 3 categorías para los vídeos, en función de si han sido catalogados, están pendientes o ya han sido retirados. También se pueden realizar búsquedas de vídeos entre dos fechas determinadas, mediante las dos cuadros de fecha disponibles en la interfaz.

A la derecha se encuentran unos botones que entran en funcionamiento cuando existe un elevado número de vídeos (más de 10) en algunas de las categorías. Estos botones paginan el número de vídeos que se muestra de cada categoría, mostrándolos de 10 en 10 para no generar una página web muy amplia (cuando haya 11 vídeos, existirán dos botones para navegar, uno entre los vídeos del 1 al 10 y el vídeo 11 en una página nueva, como se puede apreciar en el ejemplo de la **Ilustración 31**).

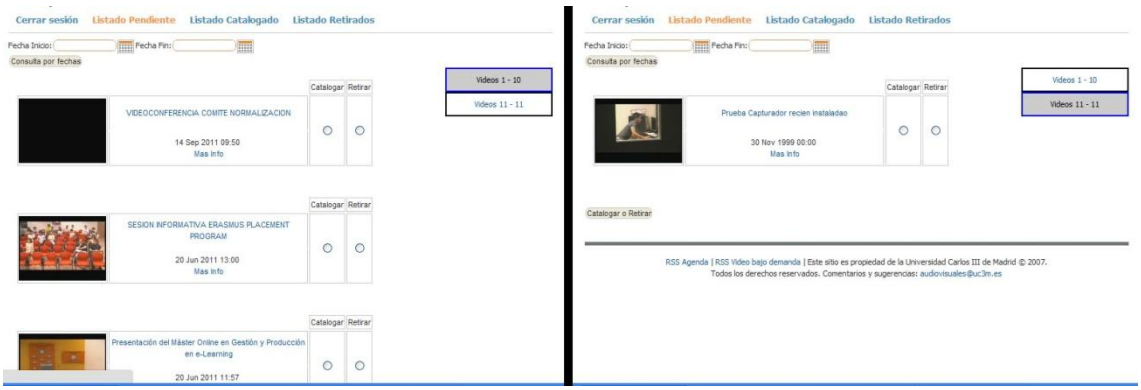


Ilustración 31 Botones para paginación de vídeos

En la parte central aparece el listado de vídeos correspondiente a la categoría seleccionada.

Ilustración 32 Recuadro básico datos del vídeo.

En la **Ilustración 32**, en la parte remarcada, figura un ejemplo de cómo se muestran los datos de cada grabación (la parte derecha se comentará en profundidad para cada una de las categorías).

A la hora de mostrar la información de las grabaciones, será conveniente insertar la misma en tablas, que posteriormente serán tratadas mediante CSS para modificar su aspecto.

La tabla básica consta de 3 elementos

- Celda que muestra una miniatura de la grabación (**Ilustración 33, (1)**).
- Celda que muestra el título de la grabación y que, además proporciona un link a la grabación (**Ilustración 33, (2)**).
- Celda que muestra la hora de grabación, además de un enlace para ampliar la información acerca del vídeo. (**Ilustración 33, (3)**).

Ilustración 33 Ejemplo Tabla Básica para datos del vídeo

Estos datos se obtienen de la base de datos de las grabaciones disponibles.

Al pulsar sobre el enlace para mostrar más información (Enlace "**Mas Info**"), muestra una nueva tabla, que por defecto se encuentra oculta, donde aparecen ordenados por columnas los datos generales del vídeo (autor, idioma, duración,) y datos técnicos (resolución, tamaño, bitrate y framerate) de las grabaciones del Presentador y de la señal VGA (**Ilustración 34**).

Ilustración 34 Recuadro información desplegado

Esta tabla con datos puede mostrarse u ocultarse simplemente con volver a hacer click en el enlace "Mas Info".

Para realizar la búsqueda de dos grabaciones realizadas entre dos fechas determinadas, se pueden utilizar los botones que hay situados en la parte derecha de los campos. Al pinchar se desplegará automáticamente un calendario en el que podremos seleccionar fechas para proceder a la búsqueda. (**Ilustración 35**).

Ilustración 35 Búsqueda de vídeos por fechas

Una vez seleccionadas las fechas de búsqueda, aparecerá el listado con los vídeos de la categoría correspondiente (Pendientes, Catalogados y Retirados), que se hayan grabado en el periodo seleccionado.

- **Implementación.**

Diseño interno

Internamente las tres secciones funcionan de forma muy similar. Como se ha explicado anteriormente, en primer lugar se verá el funcionamiento común y a continuación las particularidades que existan en cada sección.

En la aplicación se va a producir un continuo envío de información, ya sea para modificar los datos mostrados en la página o para cambiar el estado de las grabaciones realizadas en el sistema ArcaMM.

Para realizar estas operaciones PHP proporciona las opciones GET y POST que permiten enviar datos al servidor y proceder a su proceso.

GET envía información al servidor a través de la URL de la página, con lo que la información enviada puede ser visualizada en la URL que solicita la información (esta información es posible protegerla fácilmente, sin que implique ningún riesgo de seguridad) mientras que POST utiliza las llamadas STDIO para el envío de información, resultando, de esta forma, invisible al usuario.

Debido a que la plataforma ArcaMM utiliza fundamentalmente variables POST, se adopta la decisión de utilizar, en la mayor parte de la aplicación estas variables POST, para asegurar una correcta integración con la plataforma.

El uso de variables POST en la aplicación tiene una doble finalidad, por un lado solicitar los vídeos de una de las categorías que se encuentren comprendidos entre dos fechas y por otro modificar el estado de los vídeos, ya sea para retirarlos (o bien recuperarlos) o para comenzar el proceso de catalogación en ArcaMM.

La aplicación se lanza dentro de la estructura de las páginas de ArcaMM, por lo que conservaá sus características básicas (el aspecto, los scripts utilizados, las hojas de estilo, etc.) como se puede observar en la siguiente imagen, en la que el área coloreada en gris será donde se integre esta aplicación, perteneciendo el resto de la página a la estructura propia de ArcaMM.

Ilustración 36 Estructura del sitio de ArcaMM y zona de integración de aplicación

La aplicación propiamente dicha se inicia una vez que se ha solicitado la página, mediante una consulta a la base de datos de las grabaciones realizadas para obtener todos los vídeos que cumplan la condición de la categoría actual.

En caso de que se haya realizado una búsqueda por fechas, se vuelve a cargar la página actual, pero modificando los criterios de búsqueda de vídeos que realiza.

Esta modificación se establece en base a la existencia o no de variables de tipo POST. Estas variables van a indicar si se ha solicitado, o no, una búsqueda por fechas.

```
Si existe variable $_POST[fdate] entonces{
   Se busca en la base de datos todos los vídeos de la categoría
comprendidos entre dos fechas
   Se obtiene el número de vídeos.
```

```
   Se obtiene array con los datos de todos los vídeos
   Se borran las variables $_POST;
}si no hay variables en memoria{
Se busca en la base de datos todos los vídeos de la categoría
Se obtiene el número de vídeos.
   Se obtiene array con los datos de todos los vídeos
```

Tabla 25 Selector para búsqueda por vídeos (con y sin fechas seleccionadas)

En el ejemplo con pseudocódigo de la **Tabla 25,** cuando se haya solicitado una búsqueda por fechas, existirán unas variables de tipo POST que harán que se ejecute la consulta, en este caso nos devuelve los vídeos que no hayan sido catalogados ni retirados y que se encuentren comprendidos entre las dos fechas especificadas.

En caso contrario, es decir, cuando se acceda a la sección directamente, sin solicitar una búsqueda específica entre fechas, se realizará la misma consulta, sin realizar una búsqueda por fechas, generando un listado distinto del anterior.

Como se puede ver en la parte de código de la Tabla 25, tras realizar la consulta se obtiene, por un lado, el número de vídeos que hay en esa categoría (con la sentencia `$numvídeos = mysql_num_rows($SacandoDatos);)` útil para realizar posteriormente los cálculos necesarios para una correcta visualización, y por otro lado, un *array* con los datos de los vídeos almacenados en dicha base de datos.

```
for ($i=0;$i<$numvídeos;$i++){
            $Datos = mysql_fetch_assoc($SacandoDatos);
            $Vídeos[$i]=$Datos;
      }
```

Tabla 26 Generación de Array de datos de vídeos

Debido a que ya se han almacenado los vídeos en un array, para mostrarlos únicamente es necesario ir recorriendo el mismo e identificar los datos que se necesiten.

Al principio de la página, en cualquiera de las 3 secciones, aparecen dos calendarios que permiten la búsqueda de vídeos entre dos fechas determinadas.

Estos campos se encuentran dentro de un formulario tipo POST, en el cual, tras introducir las dos fechas elegidas, se recarga la página con estas variables, provocando que se ejecute la consulta de búsqueda de vídeos entre ambas fechas (Parte superior del código mostrado en la **Tabla 25**).

```
<form action="./opencast.php?amod=listadopp" method="post"
class="dates">
Fecha Inicio:<input class="search3" type="text" name="fdate"
ID="fdate" maxlength="10" value="" /><img ID="fdbutton"
style="vertical-align: middle;" src="images/calendar.gif" />
Fecha Fin:<input class="search3" type="text" name="sdate"
ID="sdate" maxlength="10" value="" /><img ID="sdbutton"
style="vertical-align: middle;" src="images/calendar.gif" /><br
/>
<input type="submit" value="Consulta por fechas" />
</form>
```

Tabla 27 Formulario para búsqueda por fechas

Puesto que los vídeos se organizan en pestañas, en un máximo de 10 en cada pestaña, es necesario generar dinámicamente el número de pestañas necesarias.

Para obtener este número se utiliza la función *ceil()* de PHP, que devuelve el redondeo por arriba del número de vídeos a mostrar dividido entre 10 (si existen 36 vídeos, esta función devolverá 4, que será el número de botones y pestañas necesarias).

```
$y= ceil($numvídeos/10);
Crear las etiquetas para hacer listas
     Inicializar z=1
     Mientras i sea menos que y, entoncs
           Abrir etiqueta de elemento de lista y enlace
           Escribir el valor z y -.
           Si z+9 es mayor que el numvídeos,
                Escribir nuymvídeos
           En caso contrario
                 Escribir z+9
           Sumar 10 al valor de  z
           Cerrar el elemento de lista y el enlace
Cerrar la lista
```

Tabla 28 Generación de botones selectores de pestañas

Una vez realizada la generación de los botones, es necesario dividir en bloques la página que se generará, para posteriormente, aplicar mediante JavaScript y CSS, el efecto de pestañas disponibles para navegación.

Para dicha opción, se utiliza el siguiente código:

```
Abrir etiqueta div.
Inicializar Z =1.
     Mientras no se llegue al número de vídeos existente:
       Si z=1
         Abrir bloque <div>
       Sumar z+1

.
.
Código generación de tablas
.
.
```

```
Si z es mayor que 10
      Cerrars bloque </div>
      Definir z=1
      Definir y como y+1
```

Tabla 29 Código Generación de paneles para pestañas

Este código genera un bloque , y va controlando iteración tras iteración, que el número de elementos (vídeos mostrados en este caso), sea igual o menor de 10.

Una vez alcanzada esta cifra, se procede a cerrar el bloque actual, reiniciar los contadores y volver a generar otro bloque , repitiéndose el proceso a continuación hasta finalizar el número de vídeos.

La muestra de datos de los vídeos, se realiza generando dos tablas por cada vídeo, una con los datos principales (imagen, fecha, título y enlace) y otra con los datos técnicos.

En cada iteración realizada por el bucle `for ($x=0;$x<$numvídeos;$x++){}` se generan estas tablas, extrayendo los datos del vídeo que corresponda mostrar del anterior array generado de vídeos, mediante la variable `$Vídeos[$x][dato a mostrar].`

De esta forma se pueden obtener todos los datos que sean necesarios e incluirlos en las tablas en las que se mostrarán los mismos.

Estas tablas incluyen las etiquetas necesarias de bloque para que posteriormente se puedan aplicar correctamente los efectos correspondientes mediante la utilización de Javascript y CSS.

```
echo "<table class=\"vopencast\">";
echo "<tr>";
echo "<td class=\"nada\"></td><td class=\"nada\"></td>";
echo "<td>Catalogar</td><td>Retirar</td>";
echo "</tr>";
echo "<tr>";
echo "<td rowspan=\"2\" class=\"img\"><img
src=\"".$Vídeos[$x][url_img]."\" /></td>";
echo "<td class=\"titulo\"><a
href=\"".$Vídeos[$x][urlvisor]."\">".$Vídeos[$x][titulo]."</a></t
d>";
echo "<td rowspan=\"2\" class=\"boton\"><input type=\"radio\"
name=\"ID\" value=\"int:".$Vídeos[$x][ID]."\"/></td>";
echo "<td rowspan=\"2\" class=\"boton\"><input type=\"radio\"
name=\"ID\" value=\"borr:".$Vídeos[$x][ID]."\"/></td>";
echo "</tr>";
echo "<tr>";
echo "<td class=\"fecha\">".fechayhora($Vídeos[$x][fecha])." <br
/>";
echo "<div class=\"links\"><a href=\"#link$x\">Mas
```

```
Info</a></div></td>";
echo "</tr>";
echo "</table>";

echo "<div class=\"pages\">";
echo "<div ID=\"link$x\">";
echo "<table class=\"infov\">";
echo "<tr><td>Autor:
".$Vídeos[$x][autor]."</td><th>Presentador</th><th>VGA</th></tr>"
;
       //si hay segundo autor se muestra el dato, si no, se
muestra una celda vacia
       if (!$Vídeos[$x][autor2]){
              echo "<tr><td></td>";
       }else{
              echo "<tr><td>Autor 2:
".$Vídeos[$x][autor2]."</td>";
       }
echo"<td>Tama&ntilde;o:
".$Vídeos[$x][widthp]."x".$Vídeos[$x][heightp]."</td><td>Tama&nti
lde;o:
".$Vídeos[$x][widthv]."x".$Vídeos[$x][heightv]."</td></tr>";
echo "<tr><td>Idioma:
".$Vídeos[$x][idioma]."</td><td>Bitrate:".(bcdiv($Vídeos[$x][bitr
atep],'1024',0))."
kbps</td><td>Bitrate:".(bcdiv($Vídeos[$x][bitratev],'1024',0))."
kbps</td></tr>";
echo "<tr><td>Duracion: ".(bcdiv($Vídeos[$x][duracionv], '60',
0))." min</td><td>Framerate: ".$Vídeos[$x][frameratep]."
fps</td><td>Framerate: ".$Vídeos[$x][frameratev]."
fps</td></tr>";
echo "<tr><td></td><td>Tama&ntilde;o:
".(bcdiv($Vídeos[$x][tamanop],'1048576',0))."
Mb</td><td>Tama&ntilde;o:
".(bcdiv($Vídeos[$x][tamanov],'1048576',0))." Mb</td></tr>";
echo "<tr><td></td><td>Codec:
".$Vídeos[$x][codecP]."</td><td>Codec:
".$Vídeos[$x][codecV]."</td></tr>";
echo "<tr><th>Audio</th><td>SampleRate:
".$Vídeos[$x][samplingrp]."</td><td>Canales Audio:
".$Vídeos[$x][canalesaudiop]."</td></tr>";
echo "</table>";
```

Tabla 30 Código Generación Tablas

Este código genera las tablas (datos generales y datos técnicos) con los datos del vídeo que correspondan en cada iteración.

Es necesario mencionar las diferencias existentes entre las tablas de cada categoría.

Mientras que las categorías *Pendientes* y *Retirados* tienen botones para realizar operaciones con los vídeos (catalogarlos, retiraros o recuperarlos), la categoría *Catalogados* simplemente dispone de la información acerca de la persona que catalogó el vídeo en ArcaMM y la fecha en la que se realizó dicha operación.

Ilustración 37 Diferencia de tablas por categoría (Pendientes, Catalogados y Retirados respectivamente)

En las categorías *Pendientes* y *Retirados,* los botones forman parte de un formulario que permite realizar las diversas operaciones que se indican en el mismo.

En estas se han utilizado elementos de tipo RadioButton que permitirán definir distintos valores que serán utilizados posteriormente para realizar la acción que proceda, por ejemplo en los vídeos *Pendientes,* retirar o catalogar, y en los de *Retirados* aparecerá la opción de recuperar vídeos.

Estas acciones se realizan mediante el atributo "*Value"* de los RadioButton. Este valor envía mediante $_POST, la acción tras lo que se realiza la operación que corresponda.

```
echo "<td rowspan=\"2\" class=\"boton\"><input type=\"radio\"
name=\"ID\" value=\"int:".$Vídeos[$x][ID]."\"/></td>";
echo "<td rowspan=\"2\" class=\"boton\"><input type=\"radio\"
name=\"ID\" value=\"borr:".$Vídeos[$x][ID]."\"/></td>";
```

```
Separar en partes el "value" obtenido de los RadioButton para
obtener la función que se realizara y la ID
Si parte del "value" es "int"
     Llamar a la función webservice, que preparara y enviara los
datos que se requieran
Si parte del "value" es "borr"
     Actualizar la base de datos marcando el vídeo como retirados
     Mostrar un mensaje de confirmación
```

Tabla 31 Ejemplo Generación Tabla con RadioButton y Pseudo-Código que ejecuta la operación seleccionada

En el caso de los botones *Recuperar* y *Retirar* la operación que se realiza es actualizar la base de datos para el vídeo seleccionado, siendo el valor del campo *Retirado* igual a 0 (vídeo disponible) o a 1(vídeo retirado) según corresponda.

Si seleccionamos en la pestaña *Pendientes* la opción *Catalogar*, se lanzará en este caso el WebService que incorporará el vídeo a ArcaMM.

Una vez se haya operado con el vídeo mediante el WebService, se devolverá el control a la aplicación que modificará el estado del vídeo a *Catalogado*, y también añadirá el

nombre de la persona que haya catalogado el vídeo y la fecha en que se realizó dicha catalogación.

Interfaz Web

Una vez generado el código fuente de la página con los códigos de cada categoría es necesario mostrarlos en pantalla para así poder seleccionar la acción a realizar con el vídeo.

El formato que presenta la página antes de aplicarle diversos efectos, sin hoja de estilo, es el que se puede observar en la **Ilustración 38**:

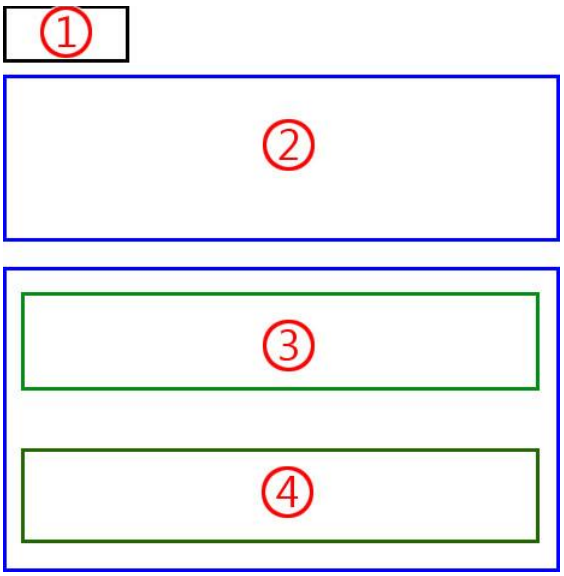


Ilustración 38 Esquema Interfaz Pasarela sin JS ni CSS

En la **Ilustración 38** se observan los siguientes elementos: **(1)** Selector de Panel de vídeos de 10 en 10, **(2)** Panel en él que se pueden mostrar los datos de hasta 10 vídeos por panel (recuadros azules) **(3) y (4)** Datos generales y datos técnicos de un vídeo (hasta 10 vídeos por panel).

Con el fin de mejorar la presentación de la página, se procede a ocultar mediante los botones del *Selector* de vídeos, todos los paneles excepto el que se encuentre activo.

Esta operación se realiza mediante una combinación de hojas de estilo y el uso de JQuery.

Mediante JQuery, se define una función donde dependiendo del botón del selector que se pulse ("`ul.tabs`" de la **Tabla 32**), se muestra el panel correspondiente (**(2) de la Ilustración 38**) y se oculta el resto.

```
$(function() {
        $("ul.tabs").tabs("div.paneles > div");
        });
```

Tabla 32 Código JS para mostrar los paneles de los vídeos

Se aplican estilos mediante CSS, con el fin de igualar la anchura que tendrán todos los paneles, y la tipografía básica que utilizarán los elementos del interior de los mismos.

Por otra parte se modifica además el aspecto del selector de paneles, trasladándolo a la parte derecha de la página, para modificar, de este modo el aspecto de los enlaces, de tal forma que aparezcan los botones y se diferencie, mediante CSS, el botón activo del selector, del resto de botones.

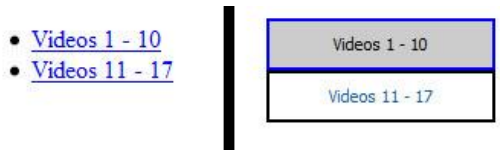


Ilustración 39 Selector de panel original y con estilos aplicados

Para la presentación de las tablas de los vídeos se realiza un proceso similar, modificando el aspecto de las tablas en las que se encuentran los datos de los vídeos, con el fin de estandarizar la presentación de datos.

Para conseguir dicho efecto se ajustaron los bordes de la tabla, editándolos para a continuación darle una mejor presentación, igualando el tamaño de las imágenes. También se editó la presentación de los enlaces presentes.

Adicionalmente, se ha ajustado el tamaño de las tablas, para que sea el mismo en todos los vídeos, independientemente del tamaño de su contenido.

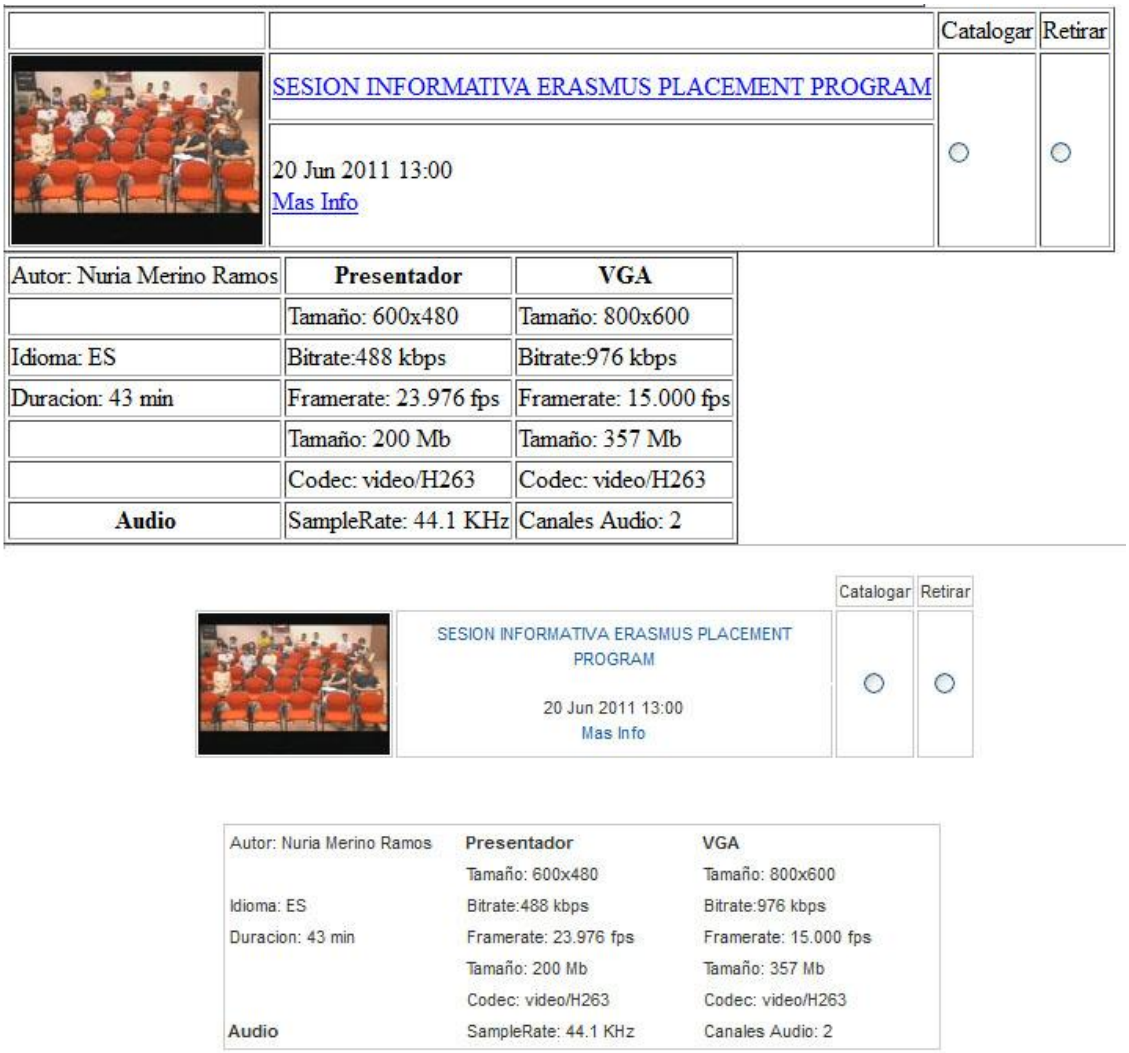


Ilustración 40 Interfaz tabla datos vídeos original y con efectos aplicados

También se adopta la decisión de ocultar la tabla de datos técnicos (ver **Ilustración 34**), de tal forma se muestre únicamente bajo petición, al hacer click en el enlace “Mas Info”.

Esta acción se realiza mediante la función *SlideToggle* de la librería JQuery, que permite mostrar u ocultar un bloque “pages” (donde se encuentran los datos técnicos de cada vídeo)

```
$(document).ready(function() {
  $('div.pages div').hide();
  $('div.links a').click(function() {
    $($(this).attr('href')).slideToggle();
    return false;
  });
```

```
});
```

Tabla 33 Codigo SlideToggle para los bloques "pages"

Este código, oculta en primer lugar todos los bloques "pages" donde se encuentran los datos (`$('div.pages div').hide();`).

En caso de que se haya realizado click en los enlaces "Mas Info", etiquetados dentro de los bloque "links", el sistema muestra u oculta el bloque correspondiente al enlace:

```
$('div.links a').click(function() {
    $($(this).attr('href')).slideToggle();
    return false;
  });
```

De esta forma se dispone en la interfaz de la información básica de cada vídeo (título, fecha y miniatura) mostrada por defecto, y los datos técnicos de cada vídeo mostrados bajo petición, sin recargar visualmente con dichos datos.

4.6. WebService

Una vez desarrollada la página de selección de vídeo, es necesario, para completar el proceso de catalogación, el envío de la grabación seleccionada a ArcaMM. ArcaMM dispone de un módulo que tras recibir información acerca de una grabación, rellena de forma automática, los datos que sean posibles de entre los enviados previamente en el módulo para añadir nuevos ítems.

En este proyecto, la tarea a realizar es el módulo que envíe información a ArcaMM, no el procesamiento de esta información.

Actualmente el método de recepción de datos de ArcaMM está basado en formularios cuyos datos son enviados mediante variables $_POST. No obstante, es posible que más adelante se decida modificar el método de envío de información, bien utilizando tecnologías SOAP o mediante otro tipo de tecnologías.

En la opción elegida se ha separado el módulo de selección de vídeo (Interfaz Web), del propio envío de datos, para que en caso de que se modificara la forma de envío, únicamente habría que modificar el funcionamiento del fichero `webservice.php`,

permaneciendo el resto de la aplicación sin cambios, sin necesidad de realizar otras modificaciones.

Para realizar esta función, es necesario partir de la acción de catalogar descrita en la sección anterior.

En caso de haber marcado un vídeo con la opción catalogar, la acción que se ejecutará es realizar una petición al fichero `webservice.php`. Este fichero es el encargado de generar el formulario a enviar a ArcaMM.

Tras realizar la llamada a este fichero, utilizando los datos de la variable $_POST[id] enviados desde la página de listados pendientes, se realiza una consulta a la base de datos que permite obtener la información acerca de la grabación que se quiere catalogar, almacenando esta información en un Array de datos denominado "Video", array en el que se puede acceder a cada uno de los datos de los que consta (extraídos de la base de datos).

```
$consulta="SELECT * FROM
vídeo,metadata,imagen,presentador,vgav,vga_descarga WHERE
metadata.id='$_POST[id]' AND imagen.id='$_POST[id]' AND
presentador.id='$_POST[id]' AND vgav.id='$_POST[id]' AND
vga_descarga.id='$_POST[id]' AND vídeo.id='$_POST[id]' AND
procesado=true;";
$SacandoDatos=mysql_query($consulta);
$Video = mysql_fetch_assoc($SacandoDatos);
```

Tabla 34 Extracción de datos para el vídeo seleccionado

Una vez obtenidos los datos se procede a generar un formulario con los datos a enviar.

Este formulario cuenta con campos ocultos que se rellenan con los datos obtenidos previamente en el array.

Para el correcto funcionamiento de esta aplicación, la dirección a la que apunte el formulario es la del módulo para añadir elementos de ArcaMM (mediante la siguiente línea que aparece en el formulario, `<form name="formopencast" action="http://bigun.uc3m.es/arcamm/manager.php" method="post">`). En esta línea se indica que los datos del formulario deben ser enviados a la dirección especificada, utilizando para ello variables tipo POST.

Para validar que los datos enviados a ArcaMM son legítimos, enviados por esta aplicación, se ha definido también un valor, una frase codificada, cuyo contenido solo es conocido por el formulario y por el módulo de ArcaMM encargado de procesar la información. De esta forma, si el formulario enviado a ArcaMM, incluye este valor clave, el contenido enviado, los datos del vídeo a catalogar serán considerados como legítimos. La sección del formulario que envía dicha información es la siguiente:

```
<input type="hidden" id="opencast" name="opencast" value="valorhash" />
```

Tras estos dos pasos fundamentales se procede a generar el resto del formulario con los datos del vídeo a catalogar.

```
echo '<input type="hidden" id=idvideo" name="idvideo"
value="'.$Video[id].'" />';
echo '<input type="hidden" id="urlvisor" name="urlvisor"
value="'.$Video[urlvisor].'" />';
echo '<input type="hidden" id="titulo" name="titulo"
value="'.$Video[titulo].'" />';
```

Tabla 35 Ejemplo generación de formulario con algunos de los datos extraídos

A pesar de que ArcaMM, actualmente solo requiere de una parte de los datos disponibles de los vídeos, (ID vídeo, URL reproductor, titulo, autor, fecha, idioma, URL imagen, altura y anchura de la imagen, datos técnicos del vídeo del presentador, URL vídeo para descarga directa, así como el tipo y el tamaño de este último), se adopta la decisión de incluir en el envío del formulario, toda la información disponible acerca de las grabaciones.

ArcaMM descartará los datos que no necesite, por lo que no supondrá ningún problema el exceso de datos enviados, ya que en caso de que en un futuro se modifique la funcionalidad de ArcaMM, requiriendo el uso de más datos, no sería necesario modificar el webservice que envía los datos desde las grabaciones de Matterhorn.

Como último parámetro del formulario se envía una URL de retorno.

Esta URL, que corresponde a la propia dirección web del formulario es enviada para que el módulo de ArcaMM conozca la URL donde debe remitir los datos del catalogador que ha validado la grabación, así como las ID del vídeo en Matterhorn y en

ArcaMM. Estos datos serán almacenados por nuestra aplicación, para la correcta visualización de la sección de vídeos catalogados.

```
<input type="hidden" id="urlvuelta" name="urvuelta"
value="http://163.117.69.141/arcamm/opencast.php?amod=webservice" />
```

Un ejemplo completo del formulario enviado seria parecido al siguiente:

```
<form name="formopencast" action="http://163.117.69.141/arcamm/webservice.php"
method="post">
<input type="hidden" id="opencast" name="opencast"
value="valorhashquedefiniremos" />
<input type="hidden" id=idvideo" name="idvideo" value="17990" />
<input type="hidden" id="urlvisor" name="urlvisor"
value="http://163.117.69.143:8080/engage/ui/watch.html?id=17990" />
<input type="hidden" id="titulo" name="titulo" value="Presentación del Máster
Online en Gestión y Producción en e-Learning" />
<input type="hidden" id="autor" name="autor" value="Carlos Delgado Kloos" />
<input type="hidden" id="fecha" name="fecha" value="2011-06-20" />
<input type="hidden" id="idioma" name="idioma" value="ES" />
<input type="hidden" id="url_img" name="url_img"
value="http://163.117.69.143:8080/static/17990/attachment-1/Presentador.jpg"
/>
<input type="hidden" id="imgheight" name="imgheight" value="480" />
<input type="hidden" id="imgwidth" name="imgwidth" value="640" />
<input type="hidden" id="duracion" name="duracion" value="1438" />
<input type="hidden" id="widthp" name="widthp" value="600" />
<input type="hidden" id="heightp" name="heightp" value="480" />
<input type="hidden" id="tamanop" name="tamanop" value="116804364" />
<input type="hidden" id="mimetypep" name="mimetypep" value="vídeo/H263" />
<input type="hidden" id="frameratep" name="frameratep" value="23.976" />
<input type="hidden" id="bitratep" name="bitratep" value="500000" />
<input type="hidden" id="samplingp" name="samplingp" value="44.1 KHz" />
<input type="hidden" id="canalesaudiop" name="canalesaudiop" value="2" />
<input type="hidden" id="widthv" name="widthv" value="800" />
<input type="hidden" id="heightv" name="heightv" value="600" />
<input type="hidden" id="tamanov" name="tamanov" value="208721018" />
<input type="hidden" id="mimetypev" name="mimetypev" value="vídeo/H263" />
<input type="hidden" id="frameratev" name="frameratev" value="15.000" /><input
type="hidden" id="bitratev" name="bitratev" value="1000000" />
<input type="hidden" id="samplingv" name="samplingv" value="44.1 KHz" />
<input type="hidden" id="canalesaudiov" name="canalesaudiov" value="2" />
<input type="hidden" id="url_descarga" name="url_descarga"
value="http://163.117.69.143:8080/static/17990/9fb65c91-76ad-45b7-9e1d-
f54deadaec9f/VGA.avi" />
<input type="hidden" id="tamanovd" name="tamanovd" value="106205126" />
<input type="hidden" id="mimetypevd" name="mimetypevd" value="vídeo/MP4V-ES"
/>
<input type="hidden" id="widthvd" name="widthvd" value="800" />
<input type="hidden" id="heightvd" name="heightvd" value="600" />
<input type="hidden" id="frameratevd" name="frameratevd" value="25.000" />
<input type="hidden" id="bitratevd" name="bitratevd" value="514099" />
<input type="hidden" id="samplingvd" name="samplingvd" value="44.1 KHz" />
<input type="hidden" id="canalesaudiovd" name="canalesaudiovd" value="2" />
<input type="hidden" id="urlvuelta" name="urvuelta" value="
http://163.117.69.141/arcamm/opencast.php?amod=webservice" />
</form>
```

Tabla 36 Ejemplo formulario a enviar completo

Una vez generado este formulario, es necesario enviarlo al servidor correspondiente. Puesto que el formulario está oculto para el usuario, es necesario encontrar una forma de envío que no requiera la pulsación manual del botón que realiza el envío.

Esta operación es realizada de forma automática, gracias al uso de Javascript, que permite realizar la función "Submit" sin necesidad de interacción alguna del usuario.

```
<script language="JavaScript">
document.formopencast.submit();
</script>
```

Tabla 37 Código para el envió del formulario de forma automática

Realizados los pasos anteriores, el formulario es enviado al módulo de ArcaMM encargado de su procesamiento.

4.6.1. Finalización de la catalogación.

Como último paso para finalizar la inclusión de un vídeo en la base de datos de ArcaMM, es necesario marcar el vídeo catalogado, en la base de datos de la aplicación.

Esta operación se realiza tras catalogar un vídeo en el módulo de ArcaMM, tras lo cual este módulo envía a la dirección especificada en el formulario, en el campo URL vuelta, mediante el uso de variable GET, los datos del catalogador que ha validado el vídeo, así como la ID de Matterhorn del vídeo y la ID de dicho vídeo en el sistema ArcaMM.

Una vez recibidos estos datos, el script anterior `webservice.php`, al detectar que recibe datos de catalogación a través de variables GET, introduce estos datos en la base de datos, actualizando el estado de la grabación a *catalogado* y añadiendo catalogador y la fecha de catalogación.

A continuación, el sistema muestra un mensaje que indica que la grabación se ha catalogado correctamente y presenta dos enlaces, uno a la dirección del vídeo en ArcaMM (utilizando para ello la ID enviada por el módulo de dicho sistema) y otro para volver a la pantalla de catalogación de vídeos de ArcaMM.

Estas operaciones son realizadas mediante el siguiente código:

```
if($_GET[editor]){
$fcatalog= date("Y/m/d H:i");
$sql=mysql_query("UPDATE vídeo SET catalogado=true,
catalogador='$_GET[editor]', fechacatalogado='$fcatalog',
idarcamm='$_GET[idarcamm]' WHERE id='$_GET[idopencast]'");
/*URL del reproductor de ArcaMM*/

//$urlreparcamm="http://arcamm.uc3m.es/arcamm/?item=";
```

```
$urlreparcamm="http://bigun.uc3m.es/svnarcamm/arcamm/?item=";

echo "<a href='$urlreparcamm$_GET[idarcamm]'>Ver en ArcaMM</a>";

echo "<br />";

echo "Video Catalogado correctamente";

echo "<br />";

echo "<a href=\"opencast.php?amod=listadop\">Volver Atras</a>";

unset ($_GET);
```

Tabla 38 Inserción datos catalogador y fecha en base de datos

5. Infraestructura utilizada.

En este capítulo se describe de manera pormenorizada las herramientas necesarias para la realización de este proyecto, tanto hardware como software, incluyendo también las herramientas adicionales necesarias.

5.1. Hardware.

- **Capturador**

  Para la implantación del capturador de Matterhorn, se ha utilizado un ordenador, con un procesador Intel i7 de doble núcleo, 2GB de memoria RAM, y un disco duro con capacidad de almacenamiento de 900 GB.

  El equipo dispone de tarjeta de sonido integrada VIA Vinyl VT1828S con 10 canales de audio disponibles, con soporte de la especificación Intel High Definition Audio.

  Se ha utilizado una tarjeta de TV marca Hauppauge, modelo WinTV-350, que permite grabar la señal de vídeo en formato MPG de forma nativa.

  También se dispone para el desarrollo del proyecto de una capturadora de señal VGA conectada al equipo por USB. La capturadora es de la marca Epiphan, modelo VGA2USB LR, capaz de grabar una señal VGA a una resolución de 1280x1024 a 60 fps.

A la hora de seleccionar la capturadora VGA se estudiaron las características de los diversos modelos para los que Matterhorn proporcionaba soporte, eligiendo el modelo VGA2USB LR, que posibilitaba la grabación de señal VGA a una resolución de 1280x1024 pixeles a 25 imágenes por segundo, apta para la grabación sin saltos, tanto de presentaciones como de vídeos.

El sistema operativo utilizado por el capturador es Ubuntu Server 10.10 de 64 bits.

- ***Core* del sistema**

Para el despliegue de los cuatro servidores Matterhorn, se han utilizado máquinas virtuales proporcionadas por la Universidad.

Estas máquinas virtuales son ejecutadas sobre la infraestructura VMWare ESX 4 (plataforma de virtualización a nivel de centro de datos) de la que dispone la propia universidad.

A estas instancias virtuales, se les ha asignado un espacio de almacenamiento de 10 GB.

Las máquinas dedicadas al procesamiento de vídeos tienen asignado el uso de dos procesadores, para ayudar en la tarea de codificar vídeos (comportamiento dual-core).

Las otras dos máquinas únicamente disponen del uso de un procesador.

El sistema operativo de estas máquinas virtuales es la versión 9.10 de Ubuntu Server, en su versión de 32 bits.

5.2. Software Servidores.

- **Apache 2**

Como requisito para la propia instalación de Matterhorn, es necesaria la instalación del servidor web Apache2.

Además, será necesario el uso de un servidor web para llevar a cabo el desarrollo del integrador con ArcaMM. Puesto que Apache2 permite el uso del lenguaje de programación PHP y la comunicación con sistemas gestores de base de datos, se toma la decisión de utilizar el mismo servidor utilizado para Matterhorn para el desarrollo y funcionamiento del integrador.

Este servidor HTTP de código abierto, es de los más utilizados y supera en número de instalaciones a otros servidores web como puede ser el IIS de Microsoft, u otras alternativas como NGINX o LightHTTP.

Entre las características de este servidor se encuentran:

- Multiplataforma.
- Modular.
- Extensible.

- **MySQL**

Debido a la cantidad de datos que maneja Matterhorn, necesita un sistema que almacene de forma segura y organizada los datos generados.

Así mismo, el integrador con ArcaMM generará una cantidad elevada de datos, relacionada con las grabaciones realizadas.

La mejor solución para mantener organizados y poder manipular una gran cantidad de datos es utilizar un sistema gestor de base de datos. En esto caso, se opta por utilizar MySQL.

Entre las características de MySQL se encuentran:

1. Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
2. Amplia variedad de tipos de datos soportados.
3. Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc).
4. Gran portabilidad entre sistemas.
5. Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.
6. Es una de las herramientas más utilizadas por los programadores orientados a Internet.
7. Infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación.
8. Gran rapidez y facilidad de uso
9. Fácil instalación y configuración.

Las razón principal por la que se ha utilizado MySQL es porque ya se disponía de su instalación para el funcionamiento de Matterhorn y por tanto no era necesario instalar de nuevo un sistema gestor de base de datos, ya que el "Extractor" también lo necesita para funcionar, aunque en caso necesario, podría haberse utilizado otro sistema como puede ser PostgreSQL.

- **Apache Maven y Apache Felix**

Apache Maven es una herramienta de software para la gestión y construcción de proyectos Java, herramienta sobre la que se ha construido Matterhorn y por tanto es obligatoria su instalación para su correcto funcionamiento.

Apache Felix es una implementación de código abierto de la especificación OSGI.

Ambas herramientas permiten el desarrollo y la ejecución de plataformas en las que interactúen varios servicios por medio de pasarelas.

El funcionamiento de Matterhorn es similar, varios servicios interactuando continuamente y por tanto necesita herramientas de este tipo para su funcionamiento.

- **PHP**

PHP es un lenguaje de programación interpretado, ejecutado en el lado del servidor y adecuado para el desarrollo web.

Este lenguaje se ejecuta en el servidor donde se ejecuta la consulta y el código enviado al cliente es el resultado de la propia ejecución, no permitiendo en ningún caso, conocer qué código fuente ha generado el resultado mostrado.

La versión utilizada es la versión 5, liberada en el año 2004. Esta es la versión más reciente en sus distintas iteraciones y la única que actualmente dispone de soporte, ya que para PHP4, el soporte finalizó en 2007.

PHP tiene múltiples modos de funcionamiento. Los utilizados en este proyecto son los siguientes.

- **Scripts en el lado servidor**: En este modo, un navegador solicita al servidor una página web. El servidor genera la información necesaria realizando las operaciones que tenga programadas y efectúa las consultas a la base de datos, tras lo que devuelve al cliente, el navegador en este caso, el resultado en HTML.

  Puesto que PHP también es capaz de procesar formularios, los formularios generados, serán posteriormente vueltos a tratar en el servidor, de la misma forma que la descrita anteriormente.

  Para que funcione este sistema es necesario instalar PHP como módulo de un servidor web, en este caso Apache 2, utilizado para este proyecto.

- **Scripts en línea de comandos**: Estos tipos de scripts generados no necesitan de interacción con el usuario para su funcionamiento.

  Estos scripts se ejecutarán de forma programada y por tanto no es necesario disponer de una interfaz para interactuar con ellos.

Por ello es necesario instalar PHP en modo línea de comandos, para que se pueda realizar la ejecución de estos scripts.

5.3. Herramientas Adicionales.

Además de las herramientas anteriormente mencionadas, durante el desarrollo del proyecto han sido necesarias otras herramientas, no se han considerado tan importantes como las anteriores, principalmente por la existencia de otras alternativas, que podrían haberse utilizado sin problema alguno.

Entre las herramientas se encuentran las siguientes:

- **MediaInfo**

MediaInfo es una herramienta que proporciona información técnica de un archivo de vídeo o audio.

Esta herramienta, tras analizar un fichero multimedia permite obtener, entre otros, los siguientes datos:

1. General: título, autor, director, álbum, número de pista, fecha, duración...
2. Vídeo: códec, aspecto, fps, bitrate.
3. Audio: códec, frecuencia de muestreo, canales, lenguaje, bitrate.
4. Texto: idioma de subtítulo.
5. Capítulos: número de capítulos, lista de capítulos.

Mediainfo es software libre, con licencia dual GPL y LGPL, disponible en versiones para diversas plataformas, como puede ser Windows, MacOS X, y también versiones específicas para distintas distribuciones Linux, como Debian, Suse, CentOS, Red Hat, Fedora.

Además dispone de versiones con interfaz visual [**Ilustración 41**] o bien interfaz por línea de comandos.

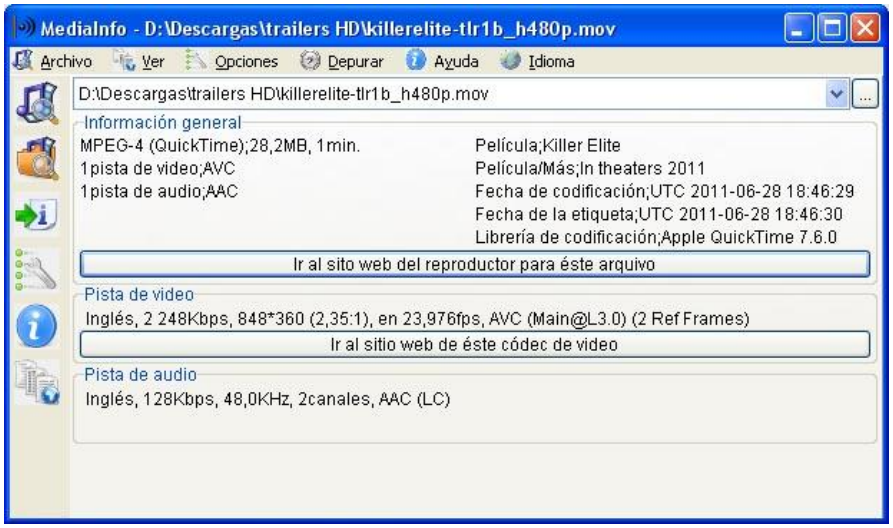


Ilustración 41 GUI de MediaInfo

Como en este proyecto, será necesario obtener datos de los vídeos cuando los scripts soliciten la información, se utilizará la versión por línea de comandos.

MediaInfo, en su versión de línea de comandos, permite mostrar toda la información relativa al fichero (ver **Ilustración 42**), en función de los datos que sean necesarios, filtrando el parámetro del dato que se quiere obtener (ver **Ilustración 43**).

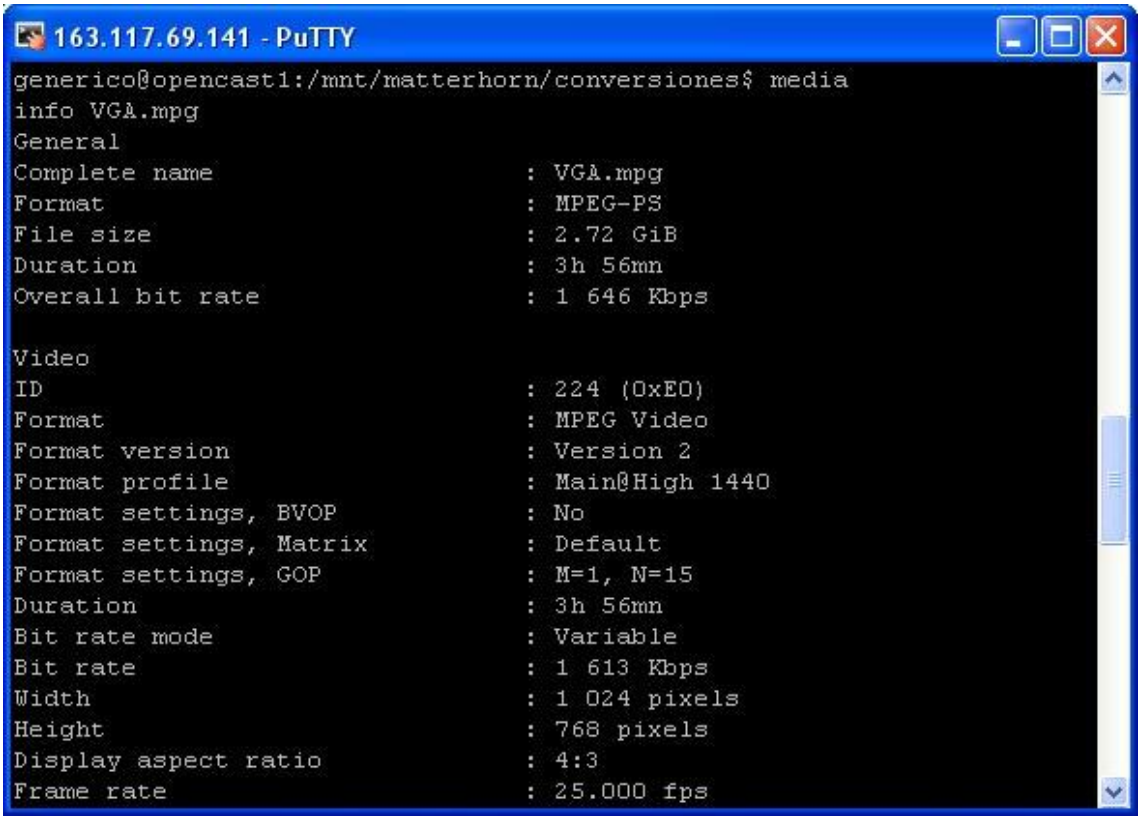


Ilustración 42 Ejemplo MediaInfo en modo CLI

Ilustración 43 Ejemplo MediaInfo CLI tras la solicitud del parámetro Duración

A la hora de desarrollar el "Extractor", se solicitará un dato en concreto del vídeo que se almacenará en una variable. Su tratamiento posterior se encuentra explicado en la sección correspondiente.

- **JQuery y JQueryTools**

JQuery es un framework para el lenguaje JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

Este framework, es muy utilizado en la denominada “Web 2.0” ya que facilita la interactuación del usuario con los sitios web de una forma que anteriormente no era posible, o lo era de una manera complicada, sin la facilidad existente en la actualidad.

Esta librería proporciona un acceso sencillo a eventos, animaciones, manipulación del árbol DOM de las páginas web, y otras varias utilidades.

JQuery Tools, es un plugin para la librería anterior, que proporciona elementos para la modificación de la interfaz de usuario creada.

Es recomendable al usar estas librerías diseñar lo mejor posible la página web que se defina, y posteriormente aplicarle efectos mediante estas librerías, aunque también es perfectamente factible diseñar una página con estas librerías como base, si bien podrían generarse problemas de accesibilidad al trabajar con distintos navegadores.

Esta circunstancia no es un problema insoluble ya que estas librerías han sido diseñadas para maximizar la compatibilidad con la mayor parte de los navegadores existentes.

En este proyecto se ha utilizado la librería JQuery Tools, que ya se utiliza en ArcaMM, por tanto no es necesario instalar ninguna otra librería.

Además proporciona la misma funcionalidad que se hubiera podido obtener con la librería JQueryUI, otra librería para efectos gráficos que también funciona sobre JQuery.

- **SimplePie**

SimplePie es una librería PHP con licencia BSD que permite leer rápidamente y parsear Feeds del tipo RSS o Atom.

Esta librería permite dado un feed RSS o ATOM válido, parsear el feed e ir seleccionando los elementos del feed uno a uno, para posteriormente utilizar dichos elementos de la forma que requiera el programador.

Los desarrolladores originales de esta librería abandonaron su desarrollo en 2009, pero el desarrollo y el soporte ha sido continuado por la comunidad, (liberando en octubre de 2011 una nueva versión), teniendo disponible en el sistema GitHub, un repositorio para la descarga y desarrollo del proyecto, y un grupo en Yahoo encargado del soporte.

Esta librería dispone de una completa API a través de la cual se puede obtener casi cualquier elemento de un feed RSS o ATOM. Dispone de funciones para extraer prácticamente cualquier dato que se halle definido en la especificación de RSS.

Esta librería proporciona además varios métodos para acceder a diversos tipos de feed entre los que se encuentran los siguientes:

- XML
- RDF
- Dublin Core v1.0 y v1.1
- MediaRSS
- GeoRSS
- iTunes

Es necesario hacer constar que los elementos de estos feeds no se pueden obtener de forma tan directa como los elementos de los feeds estándar.

La manera de actuar en estos casos es la siguiente: Dado un namespace concreto y en función del tipo de dato que se quiera obtener y una etiqueta concreta, SimplePie obtiene el contenido de dicha etiqueta, a continuación es necesario manipular el array resultante para encontrar el dato buscado.

EL proceso no presenta excesivas complicaciones, con lo que al final obtener un elemento en concreto, del tipo de feed que sea (dentro de los soportados), resulta prácticamente un proceso que no entraña dificultad alguna.

6. Desarrollo del Proyecto.

En el desarrollo de este proyecto se han investigado, desarrollado y probado diversos programas, configuraciones, lenguajes, etc, desde el inicio donde se contaba con la información del sistema que se quería utilizar, hasta la consecución del objetivo final.

El plan de acción y desarrollo de este PFC ha supuesto un proceso donde se ha investigado el funcionamiento del sistema Matterhorn, dedicando un volumen importante de tiempo a la comprensión de los diversos elementos que conforman el propio sistema, utilizando de partida la documentación disponible en el sitio del proyecto, en ocasiones escasa, para investigar las opciones de configuración, los logs del sistema, etc., buscando respuestas a cuestiones esenciales como por qué una opción no hace efecto alguno, o porque genera error esta otra opción, etc.

Este proceso por ejemplo, se generó durante la implantación del "C*ore*" del sistema y se repitió después en la instalación del agente de captura, además de tener en ocasiones que repetir el mismo proceso para actualizar el sistema a versiones de desarrollo que adolecían de cierta inestabilidad.

En la siguiente fase del proyecto, una vez que las versiones de Matterhorn alcanzaron la estabilidad y funcionalidad requerida, el objetivo era extraer más información acerca de las grabaciones de las que proporcionaban los feeds e integrar esa información en ArcaMM.

Para ello fue necesario investigar más a fondo sobre el lenguaje de desarrollo PHP, con el que se desarrollarían los scripts que aportarán los datos necesarios para el

funcionamiento del sistema, además de trabajar con Javascript y CSS para el diseño de la aplicación que sirve de pasarela.

6.1. Implementación Sistema Matterhorn.

El proceso de trabajo en este proyecto fue, en primer lugar, investigar acerca de la información disponible sobre Matterhorn en su sitio web, tanto en la web, como en la wiki disponible para tal efecto.

Posteriormente, se iniciaron conversaciones con el Servicio de Informática de la Universidad, con el fin de decidir la infraestructura que se utilizaría en la implementación de este proyecto.

Se estudiaron las dos posibilidades que proporciona el propio proyecto Opencast:

- Instalación All-In-One
- Instalación Distribuida

La versión All-In-One proporciona un entorno más sencillo de instalar y de configurar, pero penaliza el rendimiento, ya que no es lo mismo tener una máquina dedicada a cada tarea, que una sola máquina que realice las operaciones de administración, codificación y distribución, aspecto que en este caso es bastante importante puesto que se trata de un sistema en el que habrá cargas elevadas de trabajo tanto a la hora de procesar los vídeos, como a la hora de distribuirlos.

Se adopta la decisión de realizar la Instalación Distribuida utilizando servidores virtuales para la ejecución del proyecto.

Cuando se dispuso de los tres primeros servidores virtuales comenzó la configuración e instalación de los equipos.

El primer inconveniente que hubo que solventar fue la capacidad de las máquinas virtuales, en este caso se pudo apreciar que la memoria RAM disponible era insuficiente (512 MB) y no permitía la correcta instalación del sistema, por lo que se solicitó un aumento de la memoria RAM a mínimo 1GB. Esta ampliación se llevó a cabo

en los servidores 1 y 3 (*Admin* y *Engage*), mientras que en el servidor número 2 (*Worker*) aumentó la memoria a 2 GB, ya que este equipo iba a recibir una mayor carga de trabajo.

Tras esta modificación fue posible instalar la versión 1.0 de Matterhorn para así comenzar con la implantación del sistema.

Este proceso se realizó siguiendo las instrucciones del manual, aunque añadiendo en la configuración del sistema aquellos programas necesarios para el funcionamiento del sistema que no estaban incluidos en la instalación de Matterhorn

Los elementos utilizados han sido los siguientes:

- Gestor de base de datos, elección opcional entre MySQL y PostgresSQL.
- Subversion, sistema de control de versiones.
- Entorno de desarrollo de Java (JDK).
- Herramienta de software para la gestión y construcción de proyectos Apache Maven 2.

Una vez instalados y configurados estos programas, se procede a la descarga de los dos elementos esenciales necesarios, la herramienta Apache Felix (implementación OSGI sobre la que se ejecuta Matterhorn) y el propio código de Matterhorn.

Finalizado el proceso de copia de los archivos correspondientes, el siguiente paso consistió en configurar el entorno de ejecución y las variables necesarias para la compilación de Matterhorn.

Es necesario repetir este paso cada vez que se vaya a actualizar la versión a utilizar de Matterhorn.

A continuación se compila Matterhorn en los 3 servidores, utilizando para ello sus propios perfiles específicos [Ver **Tabla 1**], perfiles que dependiendo del que se vaya a utilizar, cambiarán los archivos de Matterhorn que se instalarán en cada servidor.

También es necesario utilizar en el *Admin* (servidor 1), el script DDL que creará en la base de datos la estructura necesaria para el correcto funcionamiento de Matterhorn.

Posteriormente se editan los archivos de configuración para que todos los nodos del sistema puedan comunicarse entre sí, asimismo se realizan las primeras pruebas de subida de archivos de vídeo para comprobar el ingestado de los mismos.

Quedará entonces disponible la tarjeta capturadora VGA, necesaria para poner en marcha el capturador de vídeo, tras lo que se procede su instalación en un equipo físico (Capturador).

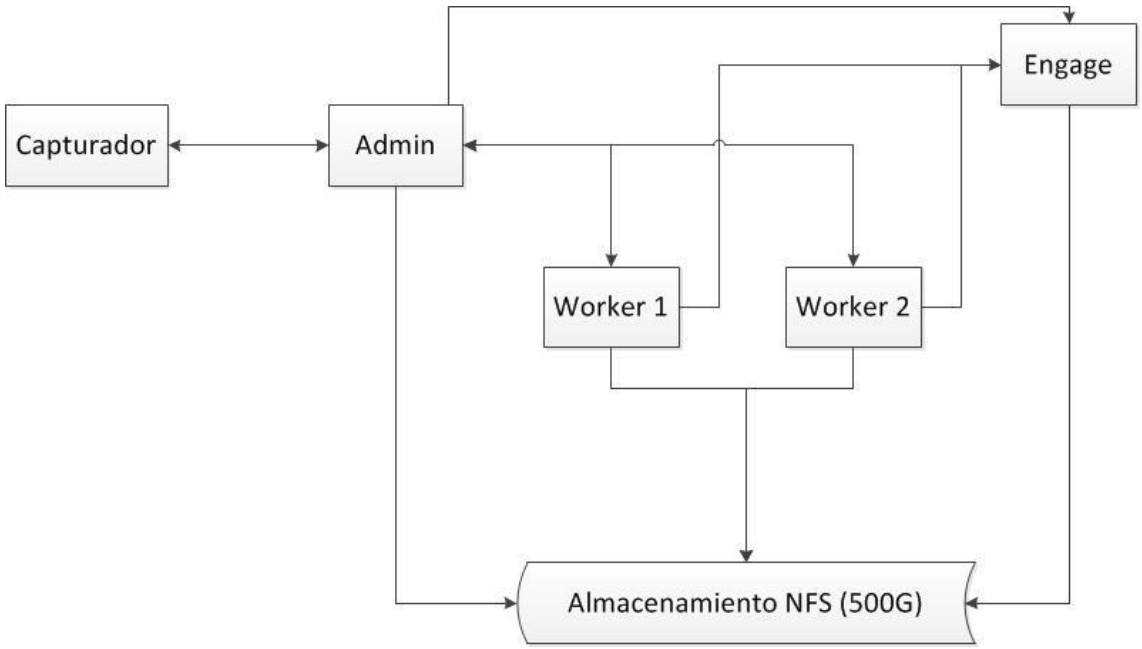


Ilustración 44 Arquitectura Servidores Matterhorn (Ver Ilustración 2)

La instalación de este equipo resulta bastante fácil, ya que es un proceso automático, en el que se deben seguir las instrucciones del script de instalación. Es necesario revisar ciertos aspectos antes de proceder a la instalación para evitar fallos en la misma. Los puntos críticos a revisar son:

1. En primer lugar es importante que los drivers de la capturadora Epiphan VGA sean los específicos para el kernel Linux que se ha estado utilizando. En caso contrario, se buscarán otros drivers válidos para el kernel en uso o un kernel compatible con dichos drivers. Es importante mencionar que este problema ha

aparecido con frecuencia, y para solucionarlo se ha optado por dejar fija una determinada versión del núcleo, ligeramente más antigua que la disponible actualmente, con el consiguiente problema de seguridad que puede aparecer, pero cuyo funcionamiento es el adecuado para el desarrollo del proyecto.

2. A continuación es necesario revisar que el sistema encargado de la grabación del sonido esté bien configurado.

   En caso de que se utilicen tarjetas de sonido integradas en la placa base del ordenador, se requiere instalar el modulo “linux-modules-backports-alsa”, para tener bien configurado el sistema de sonido, y tener disponibles todas las entradas y salidas.

   Cuando se utilice otro sistema de grabación de sonido que no sea la tarjeta de sonido integrada, es necesario instalar los drivers correspondientes para la correcta configuración de la entrada de sonido.

Una vez se hayan realizado estos pasos, se puede proseguir con la ejecución del script de instalación.

Este script va solicitando los datos necesarios, como el nombre del capturador, los dispositivos de captura que se quieren utilizar, la URL del *Core* del sistema, ruta de instalación, etc.

Una vez configurados estos parámetros, se descargarán las fuentes necesarias para su posterior compilación e instalación.

Finalizado este proceso, es necesario revisar el fichero de instalación y configurar todos los datos que falten para que el sistema funcione, durante el tiempo que dura el proceso de instalación no permite la configuración de algunos datos (por ejemplo, los datos de conexión con la base de datos, ruta donde almacenar los archivos en caso de querer utilizar otro directorio que el que aparece por defecto, etc.)

Una vez completada la configuración, y cuando el *Core* reconozca el Capturador, se realizan las primeras pruebas de captura, con grabaciones cortas y valores por defecto.

En este punto se detectan los primeros problemas del sistema, particularmente con la planificación de grabaciones, proceso que no funciona correctamente. Por el contrario, cuando se realiza una grabación no planificada, esta se realiza sin problemas.

Se encuentra la solución descargando una nueva versión del sistema que enmienda el error bug, pero como contrapartida, el servidor opencast2 (Worker 1), no interactúa correctamente con el resto del sistema, impidiendo el procesamiento de los vídeos.

En versiones posteriores del sistema se solucionan los problemas antes mencionados, pudiendo avanzar con el proyecto.

También se detectan problemas en las grabaciones de audio, ya que al principio cuando se utiliza ALSA como gestor de sonido en exclusiva, se generan problemas al capturar otras aplicaciones del sistema el gestor de audio y en ocasiones se impide la grabación del audio. Esta dificultad se soluciona más adelante utilizando PULSEAUDIO como gestor de sonido.

Con relación al *Core*, se investiga sobre el funcionamiento de los diccionarios para la extracción del texto y se añade un diccionario en castellano al sistema, para optimizar la extracción de texto.

En este punto se va actualizando prácticamente a diario el sistema con la última versión disponible de la rama (branches), lo que va proporcionando más estabilidad en el sistema.

Simultáneamente, se van realizando modificaciones, tanto en los workflows como en los perfiles de codificación, para continuar con la realización de pruebas de grabación en otros formatos y calidades diferentes de los que aparecen por defecto, como por ejemplo, grabaciones utilizando el códec H.264 o en formato WebM.

Durante este tiempo el avance del proyecto se ralentiza ya que es necesario disponer de un almacenamiento compartido vía NFS de gran capacidad que todavía no se encuentra disponible, por lo que se reduce el número de pruebas que se pueden ir realizando con el sistema, ya que no se dispone de la capacidad de disco duro necesaria.

Hay que tener en cuenta que una hora de grabación, grabada en formato MPEG-2 (formato nativo de la capturadora PCI) con las dos fuentes de vídeo, así como la de audio, puede ocupar alrededor de 1,7 GB y, teniendo en cuenta que a veces, durante el procesamiento, esta cantidad puede duplicarse o incluso triplicarse, a todas luces resulta insuficiente la capacidad de almacenamiento local disponible en las máquinas virtuales con las que se realiza el trabajo.

Durante este periodo de tiempo se libera la versión 1.0.1 (versión de mantenimiento) que soluciona la mayor parte de los problemas generados con el sistema (errores de comunicación entre los nodos, etc.), permitiendo avanzar de manera mas ágil y rápida en el PFC.

En enero se facilita el espacio disponible en un servidor NFS (500 Gb de almacenamiento) de la universidad.

Puesto que este servidor tiene el acceso restringido a ciertos segmentos de red, es necesaria la instalación en las máquinas virtuales de otra interfaz de red (virtual) con IP dentro de los segmentos permitidos para acceder al servidor NFS.

Una vez realizada esta acción, y configurados los equipos para acceder a dicho almacenamiento, se mueven desde el *Engage* todos los ficheros generados hasta la fecha, también se modifica la configuración de todos los servidores para que se direccionen hacia este servidor.

Hasta este momento, cada servidor contaba con su carpeta de configuración particular, por lo que, en caso tener que realizar cualquier modificación de configuración, era necesario operar en cada servidor por separado. Una vez dispuesto el almacenamiento compartido, se adopta la decisión de utilizar una única carpeta de configuración, moviendo dicha carpeta al almacenamiento compartido y enlazándola con los servidores a través de enlaces simbólicos.

Esta operación requiere también la modificación de los scripts de inicio de Matterhorn (que se encuentran en la ruta /etc/init.d/Matterhorn/), con el fin de añadir una

variable (la dirección IP del servidor que está utilizando cada ejecución de Matterhorn) necesaria para el correcto funcionamiento del sistema Matterhorn.

De esta forma, ya se dispone de un lugar (el disco NFS) en el que se realizarán todas las operaciones de Matterhorn, tanto el lugar de procesamiento de los vídeos, como la localización de los ficheros de distribución, vídeos de descarga, etc.

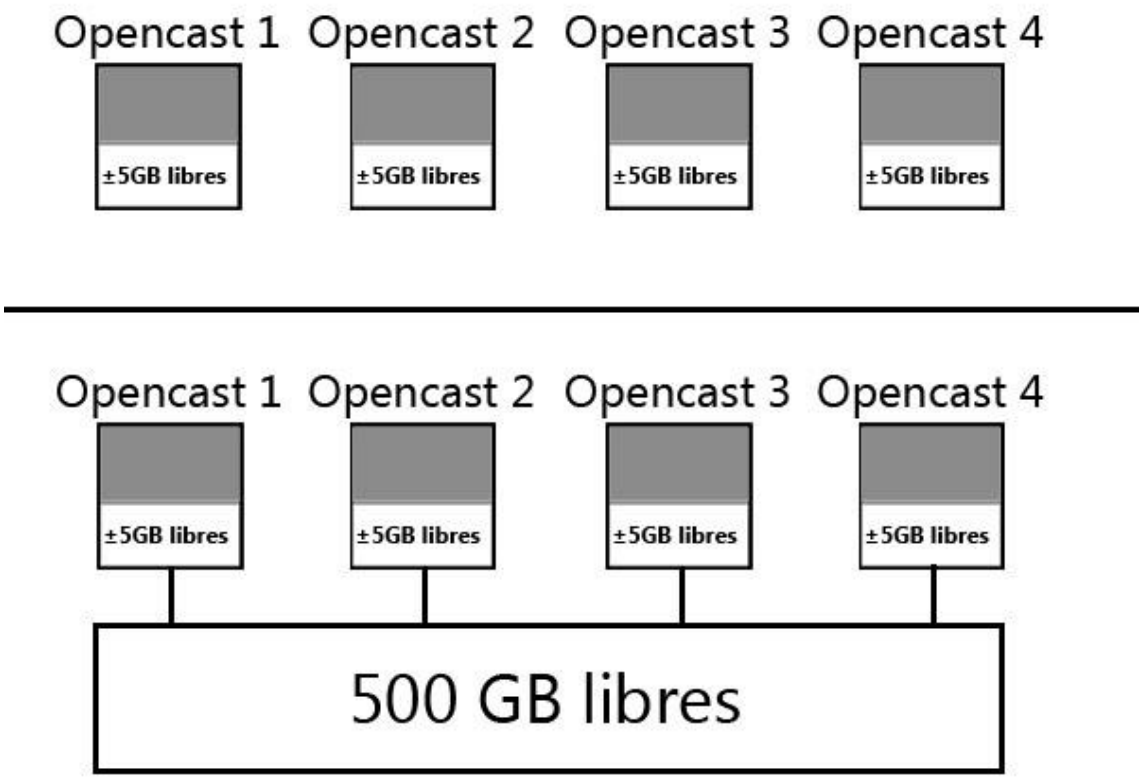


Ilustración 45 Almacenamiento sistema, antes y después de NFS

Es necesario señalar que la infraestructura utilizada en este momento por el almacenamiento compartido, presenta un inconveniente.

Se trata de un aspecto singular, puesto que para acceder al almacenamiento desde los servidores es necesario pasar por muchas máquinas, routers, etc., y también por un canal que es utilizado para otras muchas tareas, subredes de varios departamentos y envío de diferentes tipo de datos, pertenecientes a varias áreas de la universidad, con lo que el rendimiento se ve penalizado, obteniendo velocidades relativamente bajas de

transmisión de datos, sobre todo a la hora de procesar los vídeos, suponiendo un retardo importante, ya que el vídeo debe ir del sistema de ficheros NFS al servidor, para que éste lo procese y posteriormente lo devuelva al almacenamiento, aspecto que al realizarse a velocidades relativamente bajas, en archivos de gran tamaño, hace aumentar el tiempo de proceso de manera significativa.

Independientemente del aspecto anterior, ya que su mejora queda fuera del alcance de este proyecto, se continúan realizando pruebas de grabación, ahora con duraciones más largas, puesto que el espacio en disco ya no supone un problema, y se estudia, asimismo, la duración de los trabajos.

En este intervalo de tiempo la versión de desarrollo de Matterhorn 1.1, comienza a ser más estable, aunque sin ser la versión definitiva. Aún así pero se van solucionando la mayoría de los problemas que se generaban durante su desarrollo (se habían realizado pruebas anteriormente, durante una fase muy temprana de su desarrollo, pero se descartó su uso, debido a los problemas encontrados durante dichas pruebas).

Este cambio de versión implicaba un cambio bastante profundo tanto del fichero de configuración (es necesario cambiar todas las claves de configuración) como de los workflows (incorporaban entre otros aspectos, las acciones referidas a las nuevas operaciones permitidas, como el trimming de vídeo, o el subtitulado de los vídeos subidos), por lo que hubo que rehacerlos nuevo.

Esta versión soportaba entre otras cosas la reproducción de archivos F4V (archivos en contenedor de flash de alta calidad con el códec H.264). Asimismo, permitía obtener en los feeds el número de elementos que se quieren mostrar (las versiones 1.0.x solo permitían mostrar 10 elementos), siendo este aspecto esencial para la siguiente fase del proyecto, la obtención de metadatos como el título, URL, idioma o fecha para su posterior ingestado en el sistema ArcaMM de distribución de contenidos.

Durante esta fase se detecta un problema: los archivos generados no pasan de 2 GB de tamaño, acortando de este modo los vídeos.

Esto es una consecuencia provocada durante la configuración de compilación de Matterhorn, ya que según las instrucciones de la página web del proyecto, hay que utilizar la opción *workspace-stub*, que provoca que los archivos en procesamiento se envíen entre servidores a través del protocolo HTTP.

Esta opción es necesaria cuando no se dispone de almacenamiento compartido, pero tiene una limitación del protocolo HTTP, no permite enviar archivos de más de 2 GB de tamaño.

Esta cuestión se planteó en la lista de correo "Opencast-Es", cuestión que fue planteada también por otros implantadores de Matterhorn de España.

En esta fase de investigación sobre el funcionamiento de la opción *workspace-stub*, se encontró la opción correcta a utilizar, se trataba de seleccionar la opción *workspace* a la hora de definir el perfil de compilación.

Con esta opción, el sistema crea enlaces en disco desde todos los servidores, para evitar el traspaso de datos por HTTP, y de esta forma, el acceso es directo, como si se tratara de un fichero más.

Ahora ya se pueden realizar grabaciones que ocupen más de 2 GB, con la garantía de que no se verán acortados durante su procesado.

Asimismo en esta fase se encuentra la solución a un problema detectado anteriormente en la grabación de sonido, consistente en que otras aplicaciones pudieran tener cautivo el motor de sonido.

La solución fue utilizar el motor de audio PULSEAUDIO y configurarlo como "demonio" del sistema en el arranque, con el fin de que no se inicializara junto al espacio de usuario (esta acción genera problemas cuando el usuario que ejecuta la captura de grabaciones de Matterhorn es distinto del que iniciaba sesión mediante la interfaz gráfica).

Cuando finalmente se libera la versión 1.1 definitiva, que aporta mayor estabilidad respecto a la que se estaba utilizando, se observa que la funcionalidad de extracción

de texto de las diapositivas está desactivada por problemas en la detección de palabras.

Con esta versión que ya es suficientemente estable para los objetivos deseados, continúan las pruebas.

La nueva función de "trimming" que incorpora el sistema resulta bastante interesante, permite eliminar trozos de las grabaciones desaprovechados, tanto al principio como al final, y permite además la modificación de los metadatos de los vídeos.

Se procede al análisis para poder modificar tanto los workflows como el inbox para que siempre, por defecto, se realice la operación de trimming (la grabación queda a la espera de que se puedan realizar operaciones de corte y modificación de datos.).

A mediados de mayo, se decide incorporar un segundo *Worker* al sistema.

Este *Worker* es un "clon" del ya disponible, por lo que las características técnicas son las mismas.

En las pruebas realizadas con este segundo *Worker* se comprueba que durante el procesamiento de un solo vídeo, no se distribuyen las tareas de manera proporcional., uno de los workers realiza la mayor carga de trabajo mientras que el otro realiza tareas "menores", como la segmentación de vídeo o el analizador de texto.

Sin embargo, cuando hay dos o más vídeos a procesar, cada *Worker* trabaja al máximo de capacidad a la hora de procesar el vídeo, realizando diversas tareas en cada momento.

Desde las primeras pruebas se constata que al añadir más *Workers*, se ahorra tiempo durante el procesamiento de un solo vídeo y que al tener más vídeos que procesar se evita la creación de una mayor cola de procesamiento.

Esto es debido al sistema de balanceado de carga de Matterhorn, que no está diseñado para procesar diversos vídeos de una grabación en paralelo.

Las siguientes pruebas permitieron detectar un error al procesar vídeos con gran cantidad de diapositivas en el servidor de *Engage*.

En este caso era debido a un bug en la JVM, que impide que a partir de una cierta cantidad de imágenes se complete la tarea, provocando un error del workflow y resultando fallido el procesado del vídeo.

Una vez identificado el error, se detecta que se produce en las grabaciones de gran tamaño, a la hora de copiar el contenido generado a través del servidor "*Engage*", por el *bug* existente en la JVM. Y no está relacionado con el alto número de imágenes, de hecho, el error lo da un fichero de vídeo, no las imágenes en JPG.

Una vez comunicada la aparición de este error tanto a la comunidad española como a la comunidad general de Opencast, los desarrolladores del proyecto proceden al estudio del problema detectado.

Tras la liberación de algunos parches que no solucionaban este problema, finalmente se libera un parche para la versión 1.1 de Matterhorn que aporta la solución deseada.

Aplicado el parche a Matterhorn y vuelto a compilar, se repiten las pruebas anteriores, resultando esta vez que las grabaciones que antes presentaban fallos, ahora se procesan correctamente.

En una fase posterior y al objeto de dejar el sistema preparado para el siguiente curso, y por tanto para las grabaciones que se pudieran realizar a posteriori, se procede a instalar la última versión liberada de Matterhorn, la versión 1.2.

Con el fin de evitar posibles problemas de incompatibilidad con ficheros de versiones anteriores, se realiza una instalación desde cero, sin reutilizar fichero alguno.

Una de las diferencias observadas con las versiones anteriores, es que no requiere de un script DDL para la generación de la base de datos, ya que ésta es generada dinámicamente en la medida se van añadiendo los servidores al núcleo del sistema.

Esta versión que incorpora numerosas mejoras de estabilidad y rendimiento, es desplegada en un breve espacio de tiempo, quedando disponible para realizar las nuevas grabaciones del próximo curso.

Esta versión tras las comprobaciones correspondientes para asegurar su correcto funcionamiento, es la versión que se ha utilizado en el momento de finalizar este proyecto fin de carrera.

6.2. Implementación Pasarela ArcaMM.

Una vez que estuvieron operativas las primeras versiones de Matterhorn utilizadas, y tras poder acceder a los datos que proporcionan los feeds que se iban a utilizar, se dio inicio a la siguiente fase del proyecto.

En cuanto a la tecnología a utilizar, se adopta la decisión de usar para la implementación de la pasarela, el lenguaje PHP, y MySQL como sistema gestor de base de datos, ya que estas dos herramientas son las utilizadas en ArcaMM, con lo que nos aseguramos de que no se requiera de otras herramientas para su implementación

En primer lugar se definen las líneas básicas del desarrollo a seguir, consistente en la extracción de datos del feed RSS seleccionado, con el objetivo de obtener el identificador que permitirá buscar en el sistema de archivos de Matterhorn los vídeos que se necesiten, para realizar su análisis, operación tras la que se almacenarán los datos extraídos para su posterior uso en ArcaMM.

- **Manipulación de Feeds**

El primer paso a realizar es parsear el feed RSS para poder manipular los datos que lo componen de acuerdo a nuestras necesidades. Para esta acción se estudia la existencia de scripts que parsean feeds.

Entre las opciones que se barajaron, se encuentran los scripts “MagpieRSS” y “PHP Universal Feed Parser”.

Tras una serie de pruebas con estos scripts, se constata que ambos tipos de scripts se muestran excesivamente rígidos a la hora de parsear los feeds, sobre todo cuando estos no cumplen el estándar al 100%. No obstante, estos scripts disponen de opciones para que el análisis de los feeds cuando sean válidos al 100%, se efectúen de todas formas.

En cuanto a "Magpie RSS", una vez analizado el feed, guarda los campos que encuentre en cada etiqueta en arrays de datos, por lo que es necesario a la hora de extraer los datos analizar cada array por separado.

"PHP Universal Feed Parser", es un script que proporciona algunas funciones para poder acceder a diferentes elementos del feed. Sin embargo para obtener la totalidad de datos del feed, incluyendo los elementos pertenecientes a otros *namespaces*, es necesario, como en el script antes mencionado, manipular arrays, para localizar los datos necesarios.

Como tercera alternativa, se procede al estudio de la librería SimplePie (**Ver sección SimplePie, pag. 118**). Esta librería, es más laxa con los feeds que no cumplan el estándar al 100%, además de proporcionar funciones para obtener los datos de los feeds directamente, sin necesidad de manipular arrays. Además proporciona soporte a un número variado de *namespaces* que se pueden encontrar en dichos feeds. Como ventaja adicional, esta librería, sólo requiere de un fichero para su uso, el fichero *simplepie.inc.php*, que contiene todos los elementos necesarios para ejecutar este script.

Una vez establecida la librería a utilizar, se comienza la extracción de datos del feed, parseando todos los datos de los que consta, y extrayendo también los datos de los *namespaces* de Dublin Core e iTunesRSS.

- ## Desarrollo del Extractor de Datos

Tras realizar la extracción de todos los datos del feed, se observa que la cantidad de datos es insuficiente para su posterior inserción en ArcaMM, por lo que es necesario efectuar el análisis de todo el material grabado.

Para realizar esta operación, son necesarios dos requisitos:

- Posibilidad de ejecutar comandos desde un script en PHP.
- Obtención de las rutas donde se almacena el contenido de las grabaciones.

La primera funcionalidad es solucionada mediante el uso de la función *exec()* de PHP.

Esta función, permite ejecutar comandos en el sistema de archivos donde se esté ejecutando el script, además de permitirnos guardar la salida de dicho comando.

La aplicación de esta función al presente proyecto sirve para ejecutar el programa MediaInfo (**ver pág. 115**) para los archivos que haya que analizar, junto con el parámetro que indica el dato que se quiere obtener, para poder guardar el valor que se obtiene de vuelta.

En cuanto al segundo aspecto a tener en cuenta, la ruta donde se almacenan las grabaciones, (ruta que es esencial para poder realizar el análisis anteriormente mencionado) resulta sencillo de obtener una vez que a través del feed RSS se ha obtenido la ID del vídeo que se ha analizado.

En el lugar de almacenamiento donde se encuentran los vídeos, todos los archivos de la grabación se encuentran en subcarpetas identificadas con la ID del vídeo, con lo que para obtener la ruta real de la grabación basta con añadir a la ruta de almacenamiento de los vídeos la ID de la grabación.

Una vez localizado el almacenamiento de los vídeos, se realizan pruebas de extracción de datos con MediaInfo, definiendo manualmente la ruta y los ficheros a analizar, con el objeto de observar a través de un script PHP como se realiza el proceso de extracción de datos.

Todas las operaciones que requieran el uso de comandos externos a PHP, requieren del uso de la función *exec*() que permiten ejecutar un comando externo y manipular la salida correspondiente.

A continuación, se realizarán las pruebas necesarias para poder extraer datos de los vídeos sin tener la ruta completa. Estas pruebas consistieron en lanzar una búsqueda en el sistema de archivos con el nombre que debería tener el archivo a analizar para obtener su ruta completa.

Una vez realizado este paso, se continúa la prueba lanzando la ejecución del programa MediaInfo para la ruta anteriormente obtenida, devolviendo como resultado los datos técnicos de la grabación.

Posteriormente se amplía el alcance del ensayo, utilizando el ID de cada vídeo que proporciona el feed RSS, y definiendo para cada grabación la ruta real donde se buscarán los datos de cada grabación.

Esta prueba resulta un éxito, ya que, tras definir los 3 nombres de los vídeos de que consta cada grabación, este script devuelve para todos los elementos que se encuentren en el feed RSS, los datos técnicos de cada uno de los 3 vídeos.

Una vez completada la prueba anterior, se inicia el proceso para diseñar un array de datos donde se introducirán aquellos que sea necesarios extraer, tanto del Feed RSS como los datos técnicos de los vídeos.

Este array se ha diseñado en base a los datos técnicos que necesita ArcaMM para la catalogación de las grabaciones.

Con este objeto se definen en el array tanto los metadatos que se pueden extraer del feed RSS, como las variables de almacenamiento de los datos técnicos de cada uno de los vídeos.

Para probar el correcto funcionamiento, se ejecuta dicho script mostrando el contenido del array tras cada iteración, observando que los datos aparecen correctamente, tanto los metadatos como los datos técnicos, de los elementos que componen cada grabación.

Para almacenar estos datos, es necesario realizar el diseño de la base de datos adecuado (Ver **Ilustración 28**). En función de los datos disponibles, se decide utilizar 6

tablas donde se almacenaran los datos que se extraen, una por cada uno de los vídeos generados (3), una para los datos de imagen, otra tabla de metadatos extraídos del feed, y una tabla principal, que indicará el estado de la grabación en el sistema, tambien servirá de clave principal para el resto de tablas.

Una vez creada la tabla, se procede a insertar los datos del array en la base de datos, y tras comprobar el correcto funcionamiento, constatando que se han introducido todos los datos, se procede a efectuar la limpieza del código, eliminando las partes innecesarias, las variables a mostrar etc.

El siguiente paso que dejará finalizada la parte de generación del extractor, consiste en programar el script, mediante el uso del programa "*crontab*", para que se ejecute a intervalos regulares y sin intervención del usuario.

Una vez comprobada que la planificación de ejecución del script correspondiente se ejecuta correctamente, se da por concluida la fase de creación del extractor, por lo que se puede pasar a la siguiente fase de desarrollo

- **Creación de la interfaz para ArcaMM**

Esta fase consiste en desarrollar una interfaz web para poder operar con las grabaciones con el fin de catalogarlas en la plataforma ArcaMM o poder retirarlas, a demanda.

Esta interfaz debe ir integrada dentro de la plataforma ArcaMM, por lo que en primer lugar se debe obtener la versión de ArcaMM que se utilizará para pruebas.

Una vez que se ha definido y dejado preparada la zona de la página donde se implementará la aplicación (Ver **Ilustración 36**, pag. **95**), se diseña un primer prototipo, que muestre cómo se visualizarán los datos de una grabación en las tablas utilizadas para tal efecto.

Concluido este prototipo de tabla, se amplían las variaciones posibles, incorporando la conexión con la base de datos, para que de esta forma, las tablas, sean generadas

automáticamente con los datos de los vídeos que hay almacenados en la base de datos.

A continuación, se procede a intentar separar los datos de los vídeos en pestañas, para obtener una correcta visualización cuando se inserten dentro de ArcaMM.

Con este objeto, se investiga la forma de crear pestañas con Javascript. El recurso utilizado es la documentación de la librería JQuery Tools, utilizada en ArcaMM, para encontrar la forma de crear pestañas en páginas web.

Una vez encontrada una posible forma de realización, se continúa modificando el prototipo para incorporar esta funcionalidad, operación que finaliza con resultado positivo, por lo que se procede la inserción de este prototipo en la versión de prueba de ArcaMM.

Una vez insertada correctamente, se procede a mejorar el aspecto de los botones que proporcionan la funcionalidad de las pestañas mediante CSS (ver **Ilustración 39**, pag. **102**).

Cuando se muestran los datos de las grabaciones, aparecen las dos tablas de las que están compuestas, datos generales y datos técnicos. Para mejorar el aspecto de la página, se toma la decisión de mostrar, únicamente bajo demanda, la sección de datos técnicos de cada grabación, utilizando las funciones *Slide* de la librera JQuery (SilideDown, SlideUp, SlideToggle).

En las primeras versiones implementadas, se aplicaba un evento JavaScript SlideToggle a la tabla a la que hacía referencia el enlace (se ocultaba o mostraba según se pulsara en el enlace correspondiente), al resto de tablas relacionadas se le aplicaba la función SlideUp (para ocultar las posibles tablas que hubiera desplegadas).

Esta forma de implementación era funcional, pero recargaba en exceso la página, debido a que para cada enlace, se generaban un número de eventos JS igual al número de vídeos generados, por lo que en caso de disponer de, por ejemplo 20 vídeos, se generaban en total 400 líneas de código extra (20 líneas generadas por cada código x

20 vídeos en total), lo que implicaba que con un número mayor de vídeos, se crearan muchas más líneas de código que ralentizaban el funcionamiento de la página.

Esta funcionalidad se modificó utilizando los denominados manejadores de JavaScript, que permiten modificar en vivo la estructura DOM de una página web.

Esta acción permitió que en vez de definir manualmente los eventos JS, se generen estos eventos mediante llamadas a los manejadores, en función del estado en que se encuentre el enlace donde se pulse. De esta forma, se consigue obtener la misma funcionalidad antes mencionada, utilizando únicamente para ello 6 líneas de código JavaScript (Ver **Tabla 33**, pag. **92**).

Concluido este paso, y después de comprobar en una página de ejemplo que se muestran todos los datos de los vídeos generados, con la tabla de los datos técnicos oculta por defecto, se procede a implementar la opción que realiza una búsqueda entre dos fechas determinadas de los vídeos almacenados.

El primer paso para poder realizar esta operación es insertar unos campos que permitan insertar las fechas entre las que se realizará la búsqueda de los vídeos.

En estos campos se añade un calendario desplegable que se encuentra disponible en otras secciones de la plataforma ArcaMM, por lo que se procede a reutilizar el código existente para dicho cometido.

Para que se realice una búsqueda tras seleccionar dos fechas, es necesario modificar el código de la página, para que cuando se hayan seleccionado las dos fechas, se realice la consulta a la base de datos añadiendo como parámetros de búsqueda las fechas seleccionadas. Es necesario modificar el formato de las fechas para que coincidan con el formato que acepta la base de datos MySQL.

A continuación se hace necesario dividir el listado actual en 3 listados distintos. Uno para cada categoría de las existentes (Pendientes, Retirados y Catalogados). Para ello, una vez creadas las 3 páginas correspondientes, se modifica la consulta a la base de datos de cada una de ellas, para que la búsqueda la realice en función de los valores de los campos: procesados, catalogado o retirado de la base de datos.

Tras realizar este paso, ya están disponibles las 3 páginas requeridas, que aunque muestran información distinta, en función de lo valores almacenados, disponen de la misma funcionalidad: muestra de vídeos almacenados por categoría y búsqueda de vídeos por fecha entre los correspondientes a toda categoría.

- **Implementación de funciones a realizar**

El siguiente paso es personalizar cada una de las categorías, para que realice la función requerida a cada una.

El trabajo se inicia con la sección de vídeos retirados. La funcionalidad que se requiere es poder recuperar un vídeo marcado como retirado. Esta operación se realizará añadiendo a la tabla de cada grabación un botón de tipo checkbox que incluye como valor la ID del vídeo seleccionado. Este botón, perteneciente a un formulario, cuando se seleccione uno de los vídeos retirados y se pulsa el botón de "Recuperar Vídeo", actualizará el estado del campo *Retirado* en la base de datos para el vídeo asociado de la ID correspondiente, con lo que este vídeo aparecerá de nuevo en la sección de vídeos pendientes.

Para la sección de vídeos catalogados, no se requiere el uso ni de botones ni formularios. Únicamente es necesario añadir una caja lateral a la tabla de datos del vídeo que muestre los datos relativos al catalogador del vídeo, y a la la fecha en que se realizó. Estos datos son obtenidos de la base de datos, siempre y cuando el vídeo este marcado como Catalogado.

La última sección a personalizar es la correspondiente a los vídeos pendientes. La funcionalidad que se requiere en esta sección es la de poder catalogar o retirar los vídeos.

Implementar esta funcionalidad genera un pequeño problema a la hora de tratar los formularios. En la categoría pendientes, es necesario que el formulario de selección de vídeo, incorpore 2 botones checkbox por cada vídeo, por lo que la solución aplicada en el caso de vídeos retirados, en el que el valor del checkbox era la ID del vídeo seleccionado, parámetro después utilizado a la hora de actualizar la base de datos, no

es aplicable en este caso, al disponer de dos campos, que se corresponderían con la misma ID.

La solución aplicada es añadir a cada uno de los valores del checkbox una cadena que identifique la operación que se quiere realizar tras procesar el formulario.

Las cadenas a utilizar son "`int:<id vídeo>`" y "`borr:<id vídeo>`". A la hora de procesar estos valores, en primer lugar se parsea el valor enviado, dividiéndolo en dos puntos (":") y guardando las dos partes resultantes, el prefijo que indica la operación a realizar y la ID del vídeo que se manipulará.

En caso de que el prefijo a procesar será "`borr:`", la operación que se realizará será actualizar el campo retirado perteneciente al vídeo seleccionado.

Si por el contrario el prefijo que se ha procesado es "`int:`" la operación a realizar será llamar al webservice, diseñado para tal efecto, pasando como parámetro la ID del vídeo seleccionado, para su posterior procesamiento.

Una vez realizadas las pruebas acerca el funcionamiento de la interfaz y comprobar la posible no existencia de errores, se da por finalizado el desarrollo de la interfaz que actuará de pasarela.

- **Implementación del webservice**

La última parte del proyecto consiste en remitir a ArcaMM los datos del vídeo seleccionado. Desde un principio quedó claro que a la hora de intercambiar información entre la aplicación y ArcaMM, este intercambio se realizará a través de un Webservice.

El primer paso fue definir la forma de comunicación entre la pasarela y la ArcaMM. En principio se barajó la posibilidad de utilizar el toolkit NoSOAP para el intercambio de información.

Esta posibilidad fue descartada puesto que no se iba a producir el intercambio de información entre partes de forma automática, por lo que no se estimó necesario el

utilizar este sistema. En su lugar se utilizará un sistema de intercambio información basado en el uso de formularios.

Una vez que en la interfaz de la pasarela se selecciona un vídeo a catalogar, el script `webservice.php` genera, tras obtener todos los datos de la grabación seleccionada, un formulario con campos ocultos, pero rellenados en base a los datos obtenidos, con identificadores para cada valor recuperado, que son enviados de forma automática a ArcaMM mediante el uso de variables $POST.

Tras una serie de pruebas se comprueba la correcta generación del formulario a enviar y la inserción automática de los campos a rellenar en el gestor de ArcaMM.

Como último paso, queda procesar la respuesta de ArcaMM a esta plataforma.

Esta respuesta se realizará mediante el uso de variables GET, por lo que se implementa el código que tras recibir la confirmación de catalogación de ArcaMM, añada al sistema la información del editor que ha aprobado el vídeo, la fecha en que se realizó esta operación, junto con el identificador del vídeo en ArcaMM.

A pesar de estar completamente definida la estructura de la respuesta de ArcaMM, no se encontraba implementada aún en el sistema, por lo que las pruebas se realizaron generando URLs de respuesta con datos inventados para tal efecto, con el objeto de comprobar el funcionamiento del sistema, operación que resultó positiva.

Para finalizar, una vez que este sistema de respuesta fue implementado en ArcaMM, era necesario realizar las pruebas de funcionamiento, y realizar una prueba del funcionamiento del sistema en su totalidad.

Se programó, grabó y procesó un acto programado, y se procedió a realizar la catalogación en un sistema de prueba de ArcaMM.

En el listado de grabaciones pendientes de la pasarela a ArcaMM, se seleccionó la grabación que se había programado para la prueba y se procedió a su catalogación. Tras esta operación, se accedió automáticamente a la página de catalogación de

ArcaMM, en la que aparecieron sin necesidad de interacción alguna del usuario, los datos disponibles de la grabación y que son aceptados por el sistema.

Tras cumplimentar aquellos datos que no se han podido extraer de las grabaciones de Matterhorn, se procede a catalogar el vídeo, operación tras la cual devuelve el flujo a la página de la pasarela, donde se realiza el cambio de estado de la grabación en cuestión. En ese punto se puede comprobar cómo la grabación aparece ya en la categoría de grabaciones catalogadas, junto con el editor de dicha catalogación, la fecha en que se realizó y dos enlaces, uno al vídeo en el sistema Matterhorn, y otro al vídeo en la plataforma ArcaMM, por lo que ya se está en condiciones de afirmar que el proyecto cumple su objetivo desde el momento en que se programa una grabación con el sistema Matterhorn, hasta que la grabación aparece integrada en la plataforma ArcaMM y por tanto, cumpliendo el objetivo inicial expuesto al inicio de este proyecto, con lo que se conseguía comprobar el correcto funcionamiento generado a lo largo de su desarrollo, pudiendo dar por concluido el desarrollo del proyecto una vez constatado el resultado satisfactorio de la implantación del sistema Opencast Matterhorn y su integración en ArcaMM.

7. Presupuesto

7.1. Gastos en equipamiento

A continuación se detalla el coste del equipamiento utilizado durante la realización de este proyecto.

Opencast Matterhorn es un sistema libre, con licencia ECL 2.0 (Educational Community License) y sin coste por uso.

Por tanto, aunque el sistema propiamente dicho no tiene coste alguno, si lo tiene el equipamiento sobre el que se ejecuta el mismo, atendiendo además a que se ha utilizado en infraestructura física y virtualizada.

El coste de la infraestructura física es el siguiente:

3 elementos básicos que son: el PC (Procesador Inter Core i7, 2GB de memoria RAM, 900GB de disco duro) que realiza la función de capturador (930 €), una capturadora VGA Epiphan (890 €) y una tarjeta TV de marca Hauppauge WinTV-350, cuyo coste de imputación será de 125 €.

El cálculo del coste de las máquinas virtuales resulta más complicado. Dichas máquinas han sido proporcionadas por la infraestructura de máquinas virtuales que dispone el Servicio de Informática.

Este sistema se basa en el uso de cuchillas o blades, que disponen en su interior un completo sistema para la virtualización de equipos, constando de un determinado número de CPUs, así como de una cantidad determinada de memoria RAM.

El chasis en los que encuentran estas cuchillas permite introducir hasta 16 cuchillas.

A la hora de calcular los costes de la máquina virtual, hay que tener en cuenta el uso de los recursos de cada una de ellas. Estas máquinas están equipadas con procesadores Intel Xeon E5420a 2.50GHz. El uso del procesador no es exclusivo de la instancia de máquina virtual donde se ejecuta, sino que el uso de los recursos se asigna dinámicamente en función de las necesidades.

Sin embargo, el uso de Memoria RAM por las máquinas virtuales sí requiere asignación exclusiva, por tanto este será el parámetro utilizado a la hora de calcular el coste por máquina virtual.

En total son utilizadas 4 máquinas virtuales con 2 GB de memoria RAM cada una, lo que hace un total de 8 GB utilizados. Cada cuchilla consta de 32 GB de RAM por lo que nuestra infraestructura requiere una cuarta parte de cada cuchilla.

En base a la información facilitada por el responsable de la gestión de las máquinas virtuales, cada una de las cuchillas tiene un coste 10000 €. Por tanto, el coste de la infraestructura de máquinas virtuales es de 10000/4 = 2500€. Esta cantidad, dividida por el numero de máquinas virtuales utilizadas, genera un coste de 625 € por máquina virtual (2500/4).

Concepto	Cantidad	Precio Unitario	TOTAL UNIDAD
PC Core i7 860 a 2.80GHz, 2Gb de Memoria RAM y 900Gb almacenamiento en Disco Duro	1	930 €	930 €
Capturadora VGA Epiphan VGA2USB LR	1	890 €	890 €
Capturadora TV Hauppauge PVR-350	1	125 €	125 €

Maquina Virtual con procesador Intel Xeon E5420a 2.50GHz, 2Gb de memoria RAM, 10Gb capacidad de almacenamiento	4	625 €	2.500 €
		TOTAL	4.445 €

La suma de los componentes anteriores de un coste aproximado de la infraestructura utilizada es de 4445 €

7.2. Gasto en capital humano

El gasto en capital de humano, durante el desarrollo de este proyecto, resulta más difícil de cuantificar.

Durante el desarrollo de este proyecto se han alternado épocas de baja actividad en acciones relativas al proyecto, con otras épocas con mayor carga de trabajo.

Las temporadas en las que no se realizaban cargas significativas resultaban ser aquellas en las que teniendo finalizado la sección del proyecto en la que está trabajando, no se encontraban disponibles los elementos necesarios para continuar, ya fuera porque se estuviera esperando una nueva versión de Mattehorn, o bien porque se estuviera a la espera de que se incorporaran las nuevas funcionalidades que se precisaban en ArcaMM.

Por esta causa no es posible calcular el tiempo dedicado a la realización del proyecto en base a lo que sería una planificación estándar. En su lugar, se ha efectuado un el cálculo del tiempo dedicado de forma aproximada, teniendo en cuenta que en ocasiones se ha dedicado jornadas enteras, y otras a tiempo parcial.

Con los supuestos anteriores se han calculado 8 meses de trabajo, y una media de 4 horas de trabajo diario, dando como resultado 8 x 20 = 160 días trabajados a 4 horas diarias 160 x 4 = 640 horas dedicadas al proyecto. Como el que el coste estimado por hora de trabajo es de 25 €/hora, se obtiene que el resultado del capital humano estimado para este proyecto es de 16000 €

7.3. Gastos totales.

Con la suma de los dos costes anteriormente mencionados, el presupuesto final estimado para la realización del presente proyecto es el siguiente:

PRESUPUESTO DE PROYECTO			
1.- Autor:	Christian Cotillas Palomo		
2.- Departamento:			
3.- Descripción del Proyecto:			
- Titulo	**OPENCAST-MATTERHORN: UNA SOLUCIÓN ABIERTA PARA LA CAPTURA DE CLASES Y SU IMPLANTACIÓN EN DENTRO DE LA INFRAESTRUCTURA DE LA UC3M**		
- Duración (meses)	**14**		
- Tasa de costes Indirectos:		**20%**	
4.- Presupuesto total del Proyecto (valores en Euros):			
20322	Euros		
5.- Desglose presupuestario (costes directos)			
PERSONAL			

Apellidos y nombre	**N.I.F.**	**Categoria**	**Dedicación (hombre mes)**	**Coste hombre mes**	**Coste (Euro)**	**Firma de conformidad**
Cotillas Palomo, Christian		Ingeniero Técnico	4,87	3.281,25	16.000,00	
		Hombres	4,87	**Total**	16.000,00	

EQUIPOS					
Descripción	**Coste (Euro)**	**% Uso dedicado**	**Dedicación (meses)**	**Periodo de**	**Coste Imputable**
Tarjeta Capturador	890,00	100	14	60	207,67
PC Capturador	930,00	100	14	60	217,00
Maquinas Virtuales	625,00	400	14	60	583,33

Hauppauge WinTV-350 PVR	125,00	100	14	60	32,08
				Total	1.040,08

6.- Resumen de costes

Presupuesto Costes Totales	
Personal	16.000 €
Amortización	1.040 €
Subcontratación de tareas	0
Costes de funcionamiento	0
Costes Indirectos	3.408 €
Total	20.488 €

8. Conclusiones finales.

Este proyecto ha permitido a la UC3M incorporar una alternativa al sistema LCS MediaSite, que actualmente se está utilizando, con la ventaja de que su coste es infinitamente menor que el sistema anteriormente mencionado.

Entre las funcionalidades que proporciona este sistema, además de la posibilidad de visualizar una grabación en multistream, se encuentra poder elegir durante la reproducción, la señal de vídeo en vivo, accesos directos a las diapositivas generadas, y además como funcionalidad añadida, permite la descarga directa de un vídeo generado con la señal de VGA y la señal de audio, permitiendo visualizar la grabación de la presentación al mismo tiempo que la explicación correspondiente.

Opencast Matterhorn es un sistema que durante el último año ha experimentado un avance importante en cuanto a estabilidad y funcionalidades, la versión 1.2, es muy estable y su puesta en marcha no ha supuesto excesivas complicaciones, en comparación con la primera versión, versión con menos funcionalidad y bastante inestable, que generó muchos problemas a la hora de su despliegue.

No obstante Matterhorn todavía debe desarollar una serie de características para igualar su funcionalidad a la de otras alternativas actuales (Control de Grabación, vúmetro, uso de GUI para configuración)

También se ha observado una escasez importante de documentación disponible. La existente en la actualidad se limita prácticamente a su puesta en funcionamiento

básica, no encontrando, de forma clara, por ejemplo una sección dedicada a los problemas más frecuentes encontrados.

Por estos motivos, el estado "inestable" y funcionalidad limitada de las primeras versiones utilizadas de Matterhorn , que no mostraron la estabilidad que se requeriría a una versión final (a pesar de su numeración cuando se comenzó a trabajar con ella), supuso que el tiempo de desarrollo inicialmente previsto, experimentara un alargamiento inesperado, ya que hubo que esperar a que el equipo de Opencast Matterhorn liberara una nueva “release” que solucionara los problemas que se iban encontrando y además se añadieran nuevas funcionalidades que tenían anteriormente definidas y que resultaban fundamentales para el desarrollo de este proyecto fin de carrera.

En cuanto al módulo de integración con ArcaMM, la conclusión es que cumple el objetivo inicialmente definido de integrar las grabaciones realizadas en la plataforma Matterhorn dentro de la oferta de vídeos de ArcaMM de una forma sencilla, integrada dentro de dicho sistema, contando con las funcionalidades requeridas para poder visualizar los vídeos en función de su estado (pendientes, catalogados o retirados) a través de una interfaz sencilla.

En cuanto a los conocimientos adquiridos durante la realización de este proyecto, destacan los relacionados con la generación de vídeos destinados al streaming, y las implicaciones en cuanto a formatos, distribución, y otros aspectos relevantes a tener en cuenta, acciones que conocía un poco ya que estuve durante un año como Becario del Área de Audiovisuales en Getafe, encargándome de la grabación de los actos en las aulas habilitadas para ello.

Así mismo, he incrementado mis conocimientos acerca de las nuevas tendencias relativas a las aulas docentes, las nuevas tecnologías que se aplicarán, en resumen, la forma en la que evolucionará la docencia y las TIC’s.

Este proyecto también me ha servido para mejorar mi conocimiento acerca del lenguaje de programación PHP y su uso junto con un sistema gestor de base de datos, si bien había realizado algunos trabajos relacionados anterioridad, no disponía de los

conocimientos necesarios que he tenido que adquirir para aplicarlos en el presente proyecto.

También el desarrollo del proyecto me ha permitido aplicar los conocimientos adquiridos de diseño web (HTML+CSS+JS) en un escenario de trabajo "real" y no quedando limitados a pequeñas pruebas de concepto como había ocurrido con anterioridad.

Considero además, que la experiencia de instalar un sistema para su puesta en producción ha sido muy enriquecedora, ya que me ha supuesto una buena oportunidad a la hora de conocer ventajas e inconvenientes de desplegar una infraestructura destinada a producción, y no solo en un entorno de pruebas.

8.1. Líneas futuras

Respecto a líneas futuras aplicables al contenido de este proyecto, se deben diferenciar dos puntos. Por un lado, las líneas futuras del proyecto Matterhorn, algunas de las cuales se encuentran ya definidas e incluso muchas se encuentran en proyecto para su implementación. Y por otro lado las líneas futuras aplicables a su integración con ArcaMM.

A continuación se detallan estas líneas futuras.

- **Opencast Matterhorn**
    - Integración directa para la publicación de contenido en las plataformas iTunes y Youtube.
    - Funcionalidad para añadir comentarios a las grabaciones que complementen la presentación.
    - Implementación completa de codificación y reproducción correcta de grabaciones realizadas con el códec H.264 para el *Engage.*
    - Reproductor de vídeo desarrollado en HTML5, sin requerir para su visionado el uso de plugins de terceros (plugin Flash).
    - Mejora de las opciones interacción social.

- Incorporación de un “Monitor de previo” (o Control de Realización) para visualizar en vivo los parámetros de la grabación en curso.
- Editor de subtítulos integrados.
- Posibilidad de emitir en directo la grabación en curso.
- Mejoras del balanceo de carga del procesamiento de vídeos, permitiendo que los diversos vídeos de una misma grabación sean procesados por distintos *Workers.*

- **Integración con ArcaMM**
  - Integración de la planificación de eventos de ArcaMM con la planificación de Matterhorn, posibilitando la planificación automática de eventos.
  - Actualización de estado de la agenda de ArcaMM cuando el procesamiento de un acto grabado con Matterhorn se haya procesado por completo.
  - Integración de la herramienta de subtitulado de ArcaMM con Matterhorn.
  - Creación de estadísticas de grabaciones realizadas con OC-Matterhorn.

Anexos

Anexo 1: Ficheros de configuración

Config.properties

```
# The HTTP server port.  If you set this to port 80, you need to run Matterhorn as root.
Alternatively, if you want users to access Matterhorn on port 80 but do not want to run
as root, keep the default port (8080) and use an Apache HTTPD server with mod_proxy to
forward port 80 traffic to Matterhorn on port 8080.
org.osgi.service.http.port=8080

# Whether Matterhorn itself should handle HTTPS traffic.  Even if you set this to
'false',you can still use an Apache HTTPD server as a proxy to handle SSL)
org.osgi.service.http.secure.enabled=false

# The secure server port to use if running Matterhorn itself with HTTPS (as opposed to a
proxy handling HTTPS).
#org.osgi.service.http.port.secure=8443

# The public URL of this matterhorn installation.  If this felix installation is proxied
behind an Apache HTTPD reverse proxy, this URL should point to the proxy's port (usually
80).
org.opencastproject.server.url=http://${server}:8080

# The url of the remote service registry.  This is used in cases where there is no
direct connection to the service registry database such as capture agens running in
protected environments. This is typically true for capture agents and should be set to
the url of a server running the actual implementation of the service registry and the
path to the service registry(admin, Worker, etc. See the build profiles in pom.xml for a
complete list).
org.opencastproject.serviceregistry.url=http://163.117.69.141:8080/services

# The base URL of the server hosting the administrative tools.  If the admin tools are
deployed on this server, this should point to this server's public URL.
org.opencastproject.admin.ui.url=http://163.117.69.141:8080

# The base URL of the server hosting the engage tools.  If the engage tools are deployed
on this server, this should point to this server's public URL.
org.opencastproject.engage.ui.url=http://163.117.69.143:8080

# The base URL to use for publishing job locations. If left commented out, the local
server URL will be used.  Set this if you intend to support swapping servers with
different IPs or host names.
org.opencastproject.jobs.url=http://163.117.69.141:8080

# The url for the service registry, which will allow distributed installations to find
remote service instances.
org.opencastproject.serviceregistry.url=http://163.117.69.141:8080/services
```

```
# The directory where the system will store its processed files (including temporary
files).  This directory should be persistent between reboots (i.e., not /tmp)
org.opencastproject.storage.dir=/mnt/matterhorn/opencast/work

# The username and password to present to other Matterhorn servers when calling their
REST endpoints. The remote server must contain matching values in its config.properties.
org.opencastproject.security.digest.user=generico
org.opencastproject.security.digest.pass=pfc1011
# Optional demo account with administrative rights.
org.opencastproject.security.demo.admin.user=admin
org.opencastproject.security.demo.admin.pass=opencast
org.opencastproject.security.demo.admin.roles=ROLE_ADMIN,ROLE_USER,ROLE_OAUTH_USER


# Optional demo accounts with series memberships, but not administrative rights.  If set
to 'true', 1000 users will be loaded into the sample user directory.  Usernames are
user0, ..., user999.  Passwords are pass0, ..., pass999.
org.opencastproject.security.demo.loadusers =false

# To enable the LDAP user provider, you must uncomment this line, and run on a
Sun/Oracle JVM that provides the "com.sun.jndi.ldap" package.
#org.osgi.framework.system.packages.extra=com.sun.jndi.ldap

# The base URL of the streaming server (ususally "rtmp://<SERVER_URL>/matterhorn-
engage"). ${org.opencastproject.server.url} can not be used, because the streaming
server does not use the HTTP protocol. Streaming is not included in the default
workflow, since the Red5 streaming server is a 3rd party component that requires
separate installation.
org.opencastproject.streaming.url=rtmp://163.117.69.143/matterhorn-engage

# The directory where the matterhorn streaming app for Red5 stores the streams
org.opencastproject.streaming.directory=/opt/matterhorn/red5/webapps/matterhorn/streams

# The directory to store media, metadata, and attachments for download from the engage
tool
org.opencastproject.download.directory=${org.opencastproject.storage.dir}/downloads

# The base URL for media downloads.
org.opencastproject.download.url=${org.opencastproject.server.url}/static

# Relational Database configuration.  By default, Matterhorn uses an embedded H2
database.  A standalone database server is recommended for production systems.  If you
run the ddl script for your db vendor (see docs/scripts/ddl/) manually, (this is
recommended) set 'ddl-generation' to 'false'.
org.opencastproject.db.ddl.generation=true

# dbVendor can be any of the values listed at under the "eclipselink.target-database"
section of http://wiki.eclipse.org/Using_EclipseLink_JPA_Extensions_%28ELUG%29
 org.opencastproject.db.vendor=MySQL

# Matterhorn comes with the jdbc drivers for MySQL (com.mysql.jdbc.Driver) and
PostgreSQL (org.postgresql.Driver). To add other jdbcDrivers to the Matterhorn runtime,
rebuild the matterhorn-db module with your desired drivers.
org.opencastproject.db.jdbc.driver=com.mysql.jdbc.Driver

# The jdbc connection url, username, and password
org.opencastproject.db.jdbc.url=jdbc:mysql://163.117.69.141/matterhorn
org.opencastproject.db.jdbc.user=matterhorn
org.opencastproject.db.jdbc.pass=opencast


# Directory to store the search index.  This should be a persistent and stable directory
(default:
org.opencastproject.search.solr.dir=/mnt/matterhorn/opencast/work/searchindex

# The path to the repository of files used during media processing.
org.opencastproject.file.repo.path=/mnt/matterhorn/opencast/work/files

# The interval in milliseconds between two rounds of dispatching in the service
registry.
#org.opencastproject.serviceregistry.dispatchinterval=5000

# The base URL of the file server.  When using a shared filesystem between servers, set
all servers to use the same URL. If this is set to any value other than
${org.opencastproject.admin.ui.url}, the admin and file servers must use a single sign
on (SSO) solution such as CAS to avoid requiring users to perform multiple logins.
```

```
org.opencastproject.file.repo.url=http://163.117.69.141:8080

# The path to the working files (recommend using fast, transient storage)
org.opencastproject.workspace.rootdir=/mnt/matterhorn/opencast/work/workspace

# The ID of the default workflow definition to run when media are ingested
org.opencastproject.workflow.default.definition=perfil2

# The directory to hold the workflow service's solr configuration and data
org.opencastproject.workflow.solr.dir=${org.opencastproject.storage.dir}/workflow

# Url of the dedicated Solr server to use with the workflow service.  Note that if the
url is specified, the local workflow index as configured using
${org.opencastproject.workflow.solr.dir} will be ignored. A dedicated Solr server should
be set up in order to enable running multiple instances of the workflow service. Please
consult http://lucene.apache.org/solr/ on how to set up a standalone Solr server.
#org.opencastproject.workflow.solr.url=http://localhost:8983/solr/

# Send server configuration data to the opencast project to help us track how people are
using Matterhorn.  No security related information will be sent to the opencast project.
Comment this out to disable this feature.
org.opencastproject.anonymous.feedback.url=http://www.opencastproject.org/form/tracking

# The maximum number of concurrent files to ingest from the inbox directory
#org.opencastproject.inbox.threads=1

# The path for the ffmpeg binary in the ffmpeg encoder (default: /usr/local/bin/ffmpeg)
org.opencastproject.composer.ffmpegpath=/usr/local/bin/ffmpeg

# Configuration for the org.opencastproject.analysis.text.ocropus.OcropusTextAnalyzer
binary (default: /usr/local/bin/ocrocmd)
org.opencastproject.textanalyzer.ocrocmd= /usr/local/bin/ocrocmd

# Configuration for the org.opencastproject.inspection.impl.MediaInfoAnalyzer binary
(default: /usr/local/bin/mediainfo)
org.opencastproject.inspection.mediainfopath=/usr/local/bin/mediainfo

# The path for the qtsbtlembedder binary for QuickTime subtitle embedder (default:
/usr/local/bin/qtsbtlembedder)
org.opencastproject.composer.qtembedderpath=/usr/local/bin/qtsbtlembedder

# Email address of the server's admin.
org.opencastproject.admin.email = admin@localhost
```

org.opencastproject.org.capture.ConfiguracionManager

```
# Please note that the intervals and times specified in this file are in *seconds*
# The URL of the central core.  This assumes that all services are running on the same
machine. The above assumption might not be correct.  If so then replace the appropriate
keys below with your urls.
org.opencastproject.capture.core.url=http://163.117.69.141:8080

### Required variables start here (the agent will behave oddly without them) ###

# The URL of the caching directory under the root directory
capture.filesystem.cache.url=${org.opencastproject.storage.dir}/cache/
# The URL of the volatile directory under the root directory
capture.filesystem.volatile.url=${org.opencastproject.storage.dir}/volatile/
# The root URL where the captures should be stored prior to ingest
capture.filesystem.cache.capture.url=${capture.filesystem.cache.url}/captures/

# Image that should be displayed, if no vga-source is connected to the epiphan vga2usb.
If no image is set some color-bars are displayed
# capture.fallback.png=images/novideo.png

# The remote URL where the capture schedule should be retrieved
capture.schedule.remote.endpoint.url=${org.opencastproject.capture.core.url}/recordings/
calendars
# The time in minutes between attempts to fetch updated calendar data
capture.schedule.remote.polling.interval=5
# The local URL of the cached copy of the capture schedule
capture.schedule.cache.url=${capture.filesystem.cache.url}/schedule.ics

# Location of a centralized configuration file
```

```
capture.config.remote.endpoint.url=

# The time in seconds to wait between updating the local copy of the configuration

capture.config.remote.polling.interval=600

# The file to cache the server config, if any

capture.config.cache.url=${capture.filesystem.cache.url}/capture.properties



# The name of the agent

capture.agent.name=opencast-CA1

# The URL of the remote state service

capture.agent.state.remote.endpoint.url=${org.opencastproject.capture.core.url}/capture-
admin/agents

# The time in seconds between attempts to push the agent's state to the state service

capture.agent.state.remote.polling.interval=10

# The time in seconds between attempts to push the agent's capabilities to the state
service

capture.agent.capabilities.remote.polling.interval=10


# The URL of the remote recording state service
capture.recording.state.remote.endpoint.url=${org.opencastproject.capture.core.url}/capt
ure-admin/recordings

# Number of attempts the capture agent will attempt to ingest before waiting on the next
attempt. **/
capture.ingest.retry.limit=5
# The length of time to wait between trying to retry to ingest.
capture.ingest.retry.interval=300
# The length of time to wait until trying to ingest again after failing the number of
times in INGEST_RETRY_LIMIT.
capture.ingest.pause.time=3600

# The maximum length of a capture, defaults to 8 hours (28800 seconds)
capture.max.length=28800

# The maximum length of time in seconds to wait before force killing a capture when
stopping a recording
capture.recording.shutdown.timeout=60

# The default time in seconds between subsequent executions of the capture cleaner
capture.cleaner.interval=3600
# The default minimum available disk space, under which recordings are erased from the
system
# NOTE:  This diskspace value does *NOT* take the reserved disk space for the root user
into account
# PLEASE:  Set this value to more than the reserved space on disk (typically 5%)
otherwise the minimum disk space
#          checks will not function because there will appear to be enough disk even if
you cannot write to the 5%
capture.cleaner.mindiskspace=536870912
# The default maximum time (in days) a recording should be kept if there's enough disk
space available
capture.cleaner.maxarchivaldays=30

# confidence monitoring outputs images to this directory
capture.confidence.vídeo.location=${org.opencastproject.storage.dir}/volatile/

# enable/disable confidence monitoring
capture.confidence.enable=false
```

```
# enable/disable timestamps on confidence images
capture.confidence.debug=false

#Controls the behaviour of the agent when two scheduled events overlap or are within X
of one another.
#Setting this value to true will cause the cronologically second event to be dropped
from the schedule.
#Any other setting will have the agent shorten the second event to fit.
#Note that if the length drops below the minimum capture length then the capture will
not be scheduled.
capture.schedule.event.drop=false

#The length of time to require between capture events.  Specified in minutes.
#Note that this is a limitation of your hardware:  It takes a certain length of time for
the hardware
#to stop and then be ready to capture again.  Setting this to less than 1 will *not*
make this happen
#any faster, and will in fact cause you more problems when the agent tries to start a
second capture
#while the first is still in progress.
capture.schedule.event.buffertime=1

### Required variables end here ###


### MH-4493 Properties for Felix error watch process ###

# The following properties will define how to handle Felix is it does not stay
# running indefinitely. If, for whatever reason, Felix crashes it will not
# restart by default. The script watch_felix.sh was created to be used in
# combination with crontab as a safeguard for Felix.

# Comma-delimited list of email addresses to send notification of failure to
capture.error.emails=
# The SMTP host to send the mail from (Default: localhost)
capture.error.smtp=
# The SMTP user to send the message from (Default: current user)
capture.error.smtp.user=
# The password for the user (Default: no password necessary)
capture.error.smtp.password=
# The subject and message of the email, respectively. Use %date to put a
# timestamp in the subject/message and use %hostname to put the hostname in.
capture.error.subject="%hostname capture agent started at %date"
capture.error.messagebody="Capture agent was not running, and was just started."

### Capture device definitions ###

capture.device.Presentador.src=/dev/presentador
capture.device.Presentador.codec=ffenc_mpeg2video
capture.device.Presentador.codec.bitrate=1300000
capture.device.Presentador.codec.container=mpegtsmux
capture.device.Presentador.outputfile=Presentador.mpg
capture.device.Presentador.flavor=presenter/source
capture.device.Presentador.buffer.bytes=536870912
capture.device.VGA.src=/dev/vga
capture.device.VGA.outputfile=VGA.mpg
capture.device.VGA.flavor=presentation/source
capture.device.VGA.codec.bitrate=1300000
capture.device.VGA.framerate=15
capture.device.VGA.buffer.bytes=536870912
capture.device.AudioMH.src=hw:0
capture.device.AudioMH.outputfile=AudioMH.mp2
capture.device.AudioMH.flavor=presenter/source
capture.device.AudioMH.buffer.bytes=314572800
capture.device.names=Presentador,VGA,AudioMH
```

encoding.properties

```
####
# Profile definitions for the encoding service.
#
# In order to understand how to configure a format, take a look a the
# documentation below:
#
# profile.<format>.name:
```

```
#     Name of the format, used in download dialogs of the frontend.
#
# profile.<format>.output:
#     Type of the distribution format.
#     [vídeo|image]
#
# profile.<format>.suffix:
#     Extension that will be appended to the download.
#
# profile.<format>.mimetype:
#     Mime type used to provide proper content types.
#
# profile.<format>.input:
#     Track categories for which this format is applicable.
#     Known categories are:
#     - audio : for tracks containing only audio streams
#     - vídeo : tracks containg vídeo and probably audio
#     - enhanced-audio : for so-called enhanced audio tracks
##

# Distribution format definition for 4 by 3 flash presenter/presentation download
profile.flash.http.name = flash download
profile.flash.http.input = visual
profile.flash.http.output = visual
profile.flash.http.suffix = -15fps.flv
profile.flash.http.mimetype = vídeo/x-flv
profile.flash.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -r 15 -vcodec
flv -b 512000 -deinterlace -ab 96000 -ar 22050 #{out.dir}/#{out.name}#{out.suffix}

# Distribution format definition for 4 by 3 flash presenter/presentation download
profile.flash-vga.http.name = flash vga download
profile.flash-vga.http.input = visual
profile.flash-vga.http.output = visual
profile.flash-vga.http.suffix = -5fps.flv
profile.flash-vga.http.mimetype = vídeo/x-flv
profile.flash-vga.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -r 5 -
vcodec flv -b 512000 -ab 96000 -ar 22050 #{out.dir}/#{out.name}#{out.suffix}

4 by 3 flash presenter/presentation download
profile.flash-audio.http.name = flash audio download
profile.flash-audio.http.input = audio
profile.flash-audio.http.output = audio
profile.flash-audio.http.suffix = -5fps.flv
profile.flash-audio.http.mimetype = audio/x-adpcm
profile.flash-audio.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -ab
96000 -ar 22050 -vn #{out.dir}/#{out.name}#{out.suffix}

# Format definition for 3 by 4 flash presenter/presentation preview in Matterhorn UIs
profile.flash-preview.http.name = flash preview
profile.flash-preview.http.input = stream
profile.flash-preview.http.output = audiovisual
profile.flash-preview.http.suffix = -10fps-preview.flv
profile.flash-preview.http.mimetype = vídeo/x-flv
profile.flash-preview.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -r 10
-s 320x200 -vcodec flv -b 256000 -ar 11025 #{out.dir}/#{out.name}#{out.suffix}

# AVI mas resolucion
profile.avidl.http.name = mpeg4/avi download
profile.avidl.http.input = visual
profile.avidl.http.output = visual
profile.avidl.http.suffix = -dl.avi
profile.avidl.http.mimetype = vídeo/avi
profile.avidl.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -s 640x480 -r
25 -ar 44100 #{out.dir}/#{out.name}#{out.suffix}

# AVI mas resolucion
profile.avidl2.http.name = mpeg4/avi download
profile.avidl2.http.input = visual
profile.avidl2.http.output = visual
profile.avidl2.http.suffix = -dl.avi
profile.avidl2.http.mimetype = vídeo/avi
profile.avidl2.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -s 800x600 -
b 512000 -r 25 -ar 44100 #{out.dir}/#{out.name}#{out.suffix}

# FlashHD mas rapido y con mejor sonido
profile.flashHD.http.name = flash download
```

```
profile.flashHD.http.input = visual
profile.flashHD.http.output = visual
profile.flashHD.http.suffix = -flashhd.flv
profile.flashHD.http.mimetype = vídeo/x-flv
profile.flashHD.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -r
24000/1001 -s 600x480 -vcodec flv -b 512000 -deinterlace -ab 128k -ar 44100
#{out.dir}/#{out.name}#{out.suffix}

# FlashHD mas rapido y con mejor sonido
profile.flashHD-vga.http.name = flash vga download
profile.flashHD-vga.http.input = visual
profile.flashHD-vga.http.output = visual
profile.flashHD-vga.http.suffix = -flashdvga.flv
profile.flashHD-vga.http.mimetype = vídeo/x-flv
profile.flashHD-vga.http.ffmpeg.command = -strict inofficial -i #{in.vídeo.path} -r 15 -
s 800x600 -vcodec flv -b 1024k -ab 128k -ar 44100 #{out.dir}/#{out.name}#{out.suffix}

#Presentador en WebM a 1024*768
profile.webm1024.http.name = WebM presenter download
profile.webm1024.http.input = visual
profile.webm1024.http.output = visual
profile.webm1024.http.suffix = -presenter.webm
profile.webm1024.http.mimetype = vídeo/webm
profile.webm1024.http.ffmpeg.command = -i #{in.vídeo.path} -f webm -aspect 1.3333 -
vcodec libvpx -threads 4  -deinterlace -b 1500k -r 24000/1001 -acodec libvorbis -ar
44100 -ab 128k #{out.dir}/#{out.name}#{out.suffix}
```

org.opencastproject.analysis.vsegmenter.VídeoSegmenter.properties

```
# Configuration for the vídeo segmentation properties

# Number of consecutive identical frames in a 1 frame-per-second movie that it
# takes in order to assume a new scene (defaults to 5).

#stabilitythreshold = 5

# Percentage of pixels that may change between tow frames without considering
# them different (defaults to 0.05).

#changesthreshold = 0.05
```

Anexo 2: Referencias

APACHE, 2009, Apache HTTP Server Project. Disponible en http://httpd.apache.org/, Consultado en Mayo de 2011.

APPLE INC, 2011, Documentacion Itunes RSS, Disponible en http://www.apple.com/itunes/podcasts/specs.html, Consultado en Mayo de 2011.

CRUZ, FRANCISCO & IBÁÑEZ, NICOLÁS. ARCA: Federación de contenidos multimedia. Disponible en http://www.mundointernet.es/IMG/pdf/ponencia72.pdf, consultado en Mayo de 2011.

DUBLIN CORE, 2011, Dublin Core Metadata Model Set. Disponible en http://dublincore.org/documents/dces/ . Consultado en Mayo de 2011

FFMPEG, 2010, Documentación, Disponible en http://ffmpeg.org/documentation.html . Consultado en Diciembre de 2010

JQUERY PROJECT, 2011, API para uso de JQuery http://docs.jquery.com/Main_Page . Consultado en Mayo de 2011

JQUERY TOOLS, FLOWPLAYER, 2011, API para uso de JQuery Tools. Disponible en http://flowplayer.org/tools/index.html . Consultado en Mayo de 2011

MYSQL, ORACLE CORPORATION, 2011, Documentación. Disponible en http://dev.mysql.com/doc/ . Consultado en Mayo de 2011

OPENCAST PROJECT, 2010, Información general. Disponible en http://opencast.org/

OPENCAST MATTERHORN, 2010, Información y documentación. Disponible en http://opencast.org/matterhorn/ . Consultado desde Octubre 2010

PHP, Documentación y ejemplo de uso, Disponible en http://es.php.net .Consultado desde Mayo de 2011

RSS ADVISORY BOARD, 2009, Especificación RSS. Disponible en :http://www.rssboard.org/rss-specification . Consultado en Febrero de 2011

SIMPLEPIE, 2009, Documentacion y API de uso. Disponible en http://www.simplepie.org/ . Consultado en Mayo de 2011.

En: Opencast-ES <opencast-es@opencastproject.es>. Disponible desde Internet en: < http://opencast-es.200648.n3.nabble.com/> Consultado desde octubre de 2010

Anexo 3: Glosario

Adobe Connect	Sistema de videoconferencias multiples a través de internet, permitiendo compartir recursos (videos, presentaciones, etc).
ARCA	Agregador de Contenidos para la Comunidad Académica.
ARCAMM	Plataforma de difusión de contenidos audiovisuales de la UC3M.
ATOM	Alternativa a RSS.
AVI	Contenedor para audio y Vídeo desarrollado por Microsoft en 1992.
Binarios	Denominación que toman los ficheros de programas ya empaquetados, listos para instalar.
Canny Edge	Algoritmo de múltiples etapas para la detección de bordes en imágenes.
Cron	Programa que permite la planificación de tareas.
CSS	Hojas de Estilo en Cascada, lenguaje utilizado para modificar la presentación de sitios web.
DFXP	Formato XML para el desarrollo de subtítulos.
Dublin Core	Modelo de metadatos para contenidos en internet.
F4V	Contenedor flash para contenidos audiovisuales de alta calidad.
FFMPEG	Conversor de video libre, con soporte para muchos formatos tanto de audio como de video.
Flash	Reproductor multimedia propiedad de Adobe.
FLV	Contenedor flash para contenidos audiovisuales.
Framerate	Imágenes por segundo.
Framework	Estructura tecnológica diseñada para ayudar al desarrollo de aplicaciones, sitios webs, facilitando su desarrollo.
GUI	Siglas en ingles de Interfaz de Usuario Grafica (Graphical User Interface).
Gstreamer	Framework para complementos, flujo de datos y manejo/negociación de distintos tipos de medios audiovisuales (audio, video, codificación).
H.263	Códec de vídeo de bajo bitrate, utilizado en sus inicios para tareas de vídeo conferencia.
H.264	Códec de vídeo de alta compresión y alta calidad.
HTTP	Protocolo de transferencia de hipertexto.
Iframe	Etiqueta HTML que permite incrustar en una página web otra página distinta.
JVM	Maquina Virtual de Java. Entorno sobre el cual se ejecuta una aplicación Java.
LDAP	Protocolo que proporciona un directorio de usuario ordenado.
Moodle	Sistema de gestión de cursos de libre distribución que permite crear comunidades online.
NFS	Network File System, protocolo de red que permite a un sistema accede a ficheros remotos como si fueran ficheros locales.
NoSOAP	Framework para PHP que facilita la creación de WebServices.
OAI-PMH	Protocolo que permite el intercambio de información desde diversos proveedores y que permite búsquedas en varios repositorios de información.
OCR	Reconocimiento Óptico de Caracteres.

OSGI	Open Services Gateway Initiative, Conjunto de especificaciones que permiten desarrollar plataformas compatibles con el uso de múltiples servicios.
PCI	Interconexión de Componentes Periféricos, bus de conexión de elementos a un PC.
PIP	Siglas de Picture In Picture, tecnología que permite visualizar en una ventana dentro de un video otra grabación, de forma simultánea.
QTSubtitle	Subtitulador para videos Qucktime.
RCA	Conector eléctrico común en el mercado audiovisual.
REST	Tipo de interfaz web simple para manipular contenidos hipermedia utilizando para ello XML y HTTP, sin abstracciones adicionales a los protocolos.
RSS	Really Simple Syndication, difusión de información bajo suscripción.
SdIC	Acrónimo de Servicio de Informática y Comunicaciones.
SGDB	Sistema Gestor de Base de Datos.
Streaming	Distribución de contenido multimedia a través de la red, de forma que el producto se consuma según se descarga.
URI	Identificador Uniforme de Recurso, cadena de caracteres que identifica de forma inequívocamente un recurso en internet.
Video on Demand	Sistema de Televisión que permite el acceso a contenidos multimedia de forma personalizada.
VMWare ESX	Sistema de virtualización para instituciones ofrecido por la empresa VMWare. Inc.
VP6	Códec de vídeo utilizado en Adobe Flash Player.
W3C	Consorcio internacional que crea recomendaciones para el desarrollo de internet.
WebM	Contenedor de vídeo desarrollado por Google, utilizando para ello el códec VP8 para vídeo y el códec Vorbis para audio.
WebServices	Tecnología que permite el intercambio de información entre aplicaciones.
XML	Lenguaje de Marcado Extensible.
Xvid	Códec desarrollado de forma libre. Originalmente era la implementación libre del Códec de Vídeo DivX.

Anexo 4: Opencast Governance

Opencast Governance Approach

May 2011

The Opencast community is a collaboration of individuals, higher education institutions and organizations working together to explore, develop, define and document best practices and technologies for management of audiovisual content in academia. The community shares experiences with existing technologies and practices as well as identifying future approaches and requirements. The community seeks broad participation in this important and dynamic domain, to allow community members to share expertise and experience and collaborate in related projects.

The Opencast community also supports community-driven projects to solve common issues in management of academic audiovisual content as identified by the community. Today, the prime example is the Opencast Matterhorn project, an open source software development project to develop vídeo capture and management technologies that are of primary importance to the Opencast community's mission. The project seeks participation from committers and contributors who engage to forward effective development of the project.

The governance of the Opencast community and the Matterhorn project are interrelated, but each has unique needs driven by its objectives. This document recommends different but interrelated governance structures for the Opencast community, and for the Matterhorn project. We expect both structures to evolve, and future projects to participate in and inform this evolution. The detailed expectations and practices of the governing bodies will be documented and maintained on the Opencast Community wiki.

Collectively, the governance structure has four key purposes:

- Ensure the sustainability and health of projects and community;
- Provide the structure to support growth in the form of new adopters, collaborators, committers and resources;
- Support effective development and maintenance of Matterhorn and future projects;
- Allow both the community and the projects to achieve their goals of enhancing the management of audiovisual content in academia.

This document frames the governance structure for the Opencast community and the projects supported by the community. Today this includes Matterhorn. In order to address the full range of governance the document includes three segments:

1. Opencast community governance;

2. Project governance of those projects supported by the Opencast community, which today includes Matterhorn;

3. Coordination and review, which span both the community and the projects.

Opencast Community Governance

The community's governance structure is specifically focused on achieving broad collaboration in the academic media domain, and ensuring the long-term health of community-supported

projects. To this end, the governance approach supports broad engagement and allows significant latitude to define areas of focus that strengthen the evolving community and projects.

The community governance centers on a board elected by the Opencast Community that is designed to represent the will of the community while providing leadership to support community health.

The Board is composed of three categories of individuals, each with a unique selection process and term of service.

- Five members who are elected by the community to serve a three-year term.
- Up to two members who are selected by the board to service a one-year term based on their ability to address current needs of the community.
- One voting member that is a committer who is elected by the committers to serve a one-year term.
- On voting member that is a community evangelist who is selected by the board to serve a one-year term.

Members elected by the community are voted upon by organizations that are contributing to the community or adopting the products developed by the community. Organizations may include higher education institutions, commercial entities, education communities, or any other groups that are participating in the Opencast community.

(1) Any organization that has a current committer will be allowed two votes regardless of the number of committers they have. “Committer” status will be determined by the project governance structure of all Opencast community-supported projects and will be listed on the Opencast website.

(2) Each organization that does not have a current committer but publicly confirms that it has adopted the technology of an Opencast project will be allowed one vote. Adopting institutions will be listed on the Opencast website. Institutions that acknowledge that they have adopted an Opencast project earn the opportunity to participate in the governance process. This visibility also eases collaboration and sharing, affording greater benefit from other community organizations. Adopting organizations have full latitude to define whether they will be listed publicly and gain the associated voting rights.

Committing institutions according to (1) don’t get additional votes for adopting. Each organization may distribute its votes across board candidates in any way.

An election committee composed of the appointed board members and one external and impartial individual selected by the board will manage the election process. The election committee will hold an open nomination process for new board members sixty days prior to the election date, and communicate the nominees who have agreed to run at least 30 days prior to the election date. The election committee will also verify and communicate which organizations are eligible to vote and the number of votes each organization has earned, count all votes, communicate which board members are elected, and keep the detailed voting results confidential.

It is the responsibility of the board to ensure the current and future health of the community. This will require the board to assess the current state and identify areas of focus. Regardless of changing needs, the board has the following responsibilities.

1. Select and invite those board members that require nomination and selection by the board.
2. Effectively manage financial contributions made to the community.
3. Facilitate the community strategic planning and governance review that spans community and project governance. It is important to note that the board does not drive or own the content of the review, but is responsible for ensuring that the review occurs and for facilitating a process that effectively engages committers and community members.
4. Actively engage in drawing new organizations into the community.
5. Identify and execute processes to secure needed resources of any kind, funding, staff, etc.
6. Act on behalf of the Opencast community in creating partnerships that serve the goals of the community.
7. Actively participate in community communications activities and community building efforts.

The board will maintain both an open and a closed list for communications. The open list will include all non-confidential deliberations to allow community members to monitor and understand board activities. The closed list will be used only for sensitive and confidential discussions.

The board is not responsible for making project governance decisions, nor for dictating priorities of committers.

The primary focus areas of the board will shift over time with the evolving needs for the community. As such, board members must have a strong commitment to the success of the community and both the willingness and the ability to fill multiple roles based on the needs of the community.

Project Governance

A project's governance structure is specifically focused on achieving the vision for the product. To this end, the governance approach supports effective development processes that result in frequent, high-quality releases and provide for overall product coherence. It also allows for growth in the project, providing a clear path to contribution and influence for new committers.

Project governance centers on committers. Individuals gain influence and voting rights in the project through recognized contribution.

> We define a committer as an individual contributing to the project in a way that deserves participation in the product-related decision-making process.
>
> New committers must be nominated by a current committer, and approved through consensus as described in the governance process below.

The committers agree to the following approach to project governance:

1. Project business will be conducted on the open mail lists. Discussion and coordination may occur in meetings or other forums, but resulting recommendations will be moved to the list for broader input and decision-making.
2. Committer proposals will be voted upon with responses showing agreement (+1) and disagreement (-1). Such a vote indicates both an opinion, and a commitment to contribute to successful implementation of the proposal. Alternatively, committers may share opinions on matters where they are not agreeing to active participation with a response of agreement (+0) or disagreement (-0).
3. Consensus among all who choose to vote is required on major project decisions including but not limited to approval of new committers, changes in governance, product

releases, and development decisions that impact project goals and coherence. Consensus is defined as an absence of disagreement (or -1 votes). This definition of consensus is often referred to as lazy consensus, as an absence of disagreement is viewed as support.

4. Committers will review the project governance process twice each year to evaluate its effectiveness and refine the approach. This practice will be actively supported by the Opencast community board.
5. The project roadmap is defined as the consolidated commitments of all committers. The roadmap can be expanded, reduced or refined through a change in direct commitments from individuals or organizations.
6. All committers will allocate 20% of their project time to support the common good of the project. This expectation is a key consideration in the nomination and evaluation of new committers.
7. All committers will support release, quality assurance and testing processes for the project.

The project committers suggest the following work practices to increase the effectiveness of project governance.

- The governance process requires committers to actively monitor and participate in discussions on list. This participation is critical to maintain a healthy and vibrant decision-making process, as well as supporting new contributors to learn from the experience of the committers.
- If the discussion on list is outside of a committer's area of expertise or experience, the committer is not expected to comment or participate.

Coordinating and Review Process

The project and community governance processes require communication and coordination. To this end, the community will be called to convene twice annually. During the semi-annual meetings the community will:

- Complete a brief review of existing governance processes with the objective of validating that existing processes are serving the project and community well, or enhancing the processes to improve results.
- Share updates that span project and community status and progress.
- Review the product roadmap that is the culmination of current commitments. Discuss the project vision, bringing together the views the committers and the broader Opencast community.
- Communicate and seek to align community and development priorities.
- Capture future development and collaboration opportunities beyond the current roadmap.

The board owns the planning and facilitation of these meetings. The board does not drive or own the content of the review but is responsible for ensuring that the review occurs and for facilitating a process that effectively engages committers and community members.

All discussion and recommended decisions resulting from the coordinating and review process will follow the defined governance processes and must be communicated on list for greater participation and feedback.