UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería de Software

Trabajo Fin de Grado

VideoData: uso de vídeos para el almacenamiento de datos

Jorge Casiano Bermejo

Noviembre, 2015

UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería de Software

Trabajo Fin de Grado

VideoData: uso de vídeos para el almacenamiento de datos

Autor: Jorge Casiano Bermejo

Fdo:

Director: Antonio Polo Márquez

Fdo:

Tribunal Calificador

Presidente:

Fdo:

Secretario:

Fdo:

Vocal:

Fdo:

CLASIFICACIÓN:

FECHA:

Índice general

Capítulo 1.- Introducción

La **educación on-line** es un tipo de educación a distancia que hace uso de Internet y junto a las tecnologías de información y comunicación con el objetivo de crear un proceso de aprendizaje.

Ante el auge que ha tenido este tipo de educación surge este proyecto que pretende mejorar la educación a distancia con el desarrollo de una herramienta capaz de solucionar algunos problemas que existen todavía.

No se puede hablar de educación on-line sin mencionar a los **Massive Open Online Course** *(MOOC)* que como su nombre indica son cursos masivos, online y gratuitos. Esta modalidad en sus inicios se pensó que servirían como *herramienta* democratizadora de la educación superior. Pero presenta algunos problemas que serán analizados, proponiendo una posible solución para mejorar algunas trabas que presenta.

VideoData es un sistema integrado que permite diseñar cursos on-line de una forma sencilla y utilizando una metodología de aprendizaje más interactiva con los alumnos.

Ilustración 1: Fases del e-learning

Facilita las siguientes tareas:

1. **Preparación de actividades**

   VideoData facilita la preparación de actividades que se presentan mediante vídeos, distinguiendo la preparación de los siguientes tipos de actividades:

   - *Vídeos:* usados como material pedagógico en vez de centrar la enseñanza en contenido textual.
   - *Exámenes:* utilizados como recursos para corroborar lo aprendido

permitiendo evaluar a los alumnos además de mejorar el aprendizaje de la materia.

- *Hipervídeos:* son vídeos con integración de exámenes y otros recursos o actividades.

2. **Evaluación:**

   Una vez se dispone de una biblioteca de actividades, se seleccionan las que componen el aprendizaje del alumno que se encargará de realizarlas. La evaluación debe ser lo más rápida posible y devolver la mayor retroalimentación al alumno. Esta fase distinguiremos las siguientes tareas:

   - *Selección, despliegue y publicación:* las actividades seleccionadas son publicadas y desplegadas para que puedan ser realizadas por los alumnos.
   - *Realización de la actividad y corrección automática:* aquella parte de la actividad de tipo test será evaluada automáticamente.
   - *Corrección manual:* si existe alguna parte de la evaluación que requiera una corrección manual se realiza por el profesor o podría usarse una *evaluación por iguales (entre alumnos supervisada por el profesor)*.
   - *Guardado de resultados:* los resultados y respuestas de los alumnos son guardadas.

3. **Análisis y almacenamiento:**

   - *Análisis:* los resultados pueden ser analizados por servicios externos, el sistema está adaptado para el uso de **Flubaroo** que permite la obtención de estadísticas a partir de los resultados e incluso notificarlos por correo electrónico al alumno.
   - *Almacenamiento:* se almacenan los datos dentro de un vídeo proporcionando un almacenamiento de datos masivos, además de seguridad debido a la encriptación/codificación con el vídeo.

Objetivos

En este trabajo se han establecido una serie de objetivos principales a realizar, son los siguientes:

- Estudiar los MOOCs y sus problemas.
- Buscar una solución e implementarla.
- El sistema debe ser fácil de utilizar.
- Emplear en el sistema tecnologías aprendidas durante el grado GIIIS.
- Orientar el sistema hacia la escalabilidad.
- Centrar el sistema en la especialidad multimedia.

Estructura del documento

TODO: será el último apartado a realizar.

Introducción a VideoData: un sistema de apoyo a la enseñanza online

Para sacar el máximo rendimiento a este sistema el usuario ideal son los profesores y educadores que quieran desempeñar su función de forma online, es decir usando **e-learning o** aprendizaje electrónico que es algo que está en auge ya que permite universalizar la enseñanza.

Ilustración 2: Sistema VideoData

La aplicación contiene tres módulos que pueden ser usados de forma independientes pero toda su potencia radica en el uso de los módulos de forma unificada.

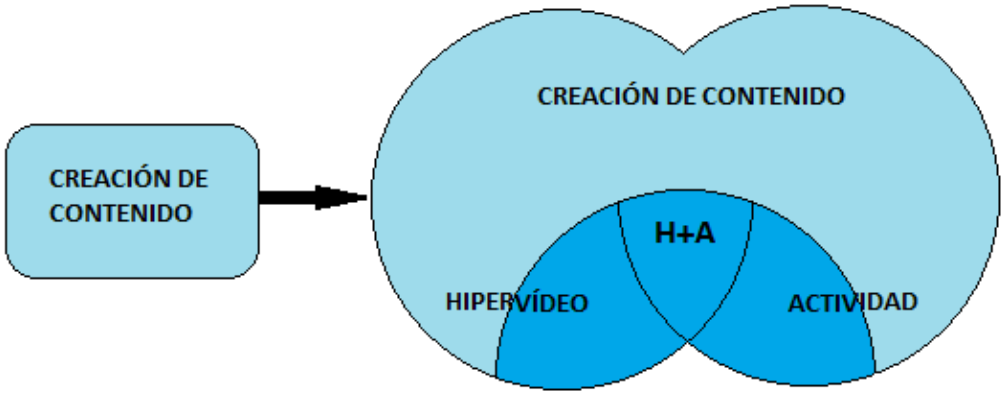


Ilustración 3: Creación de contenido

En la creación de contenidos se distinguen dos fases esenciales:

1) La fase de **diseño de contenidos** en la que se seleccionan los objetivos de aprendizaje y se diseñan los contenidos dirigidos a alcanzar esos objetivos. Es importante destacar que entenderemos:

   Contenidos = Recursos de aprendizaje + Actividades

   que representan la integración de la teoría clásica (incluida en los recursos de aprendizaje) y la parte de prácticas (incluida en las Actividades).
2) La fase de elaboración de contenidos, que están guiados por videos orientados a su uso online por el alumno, con enlaces a otros recursos (hipervideo) y en la mayoría de los casos incluyendo también a actividades integradas con el contenido del hipervideo.

Por tanto, en la **creación de contenido se utilizan los hipervídeos y las actividades** y de este modo se integrarán las actividades durante la visualización del hipervídeo.

Un tipo especial de actividad son los exámenes, que sirven para poder evaluar a los alumnos acerca del temario enseñado y, a la vez, para motivar y afianzar al alumno en la adquisición de conocimientos y competencias. El examen es un conjunto de preguntas que pueden ser de distinto tipo:

- Test simple: son preguntas que tienen distintas opciones de las cuales solo una es correcta. Estas preguntas serán corregidas de forma automática.
- Test múltiple: son preguntas que tienen distintas opciones entre las cuales puede haber varias correctas. Estas preguntas serán corregidas de forma automática.

- Desarrollo: contienen una sección para introducir una respuesta que será de tipo teórico. Este tipo de preguntas deben ser corregidas de forma manual.
- Recurso: contienen un campo para introducir un enlace al lugar donde se encuentra el recurso que responde correctamente a la pregunta. Estas preguntas deben ser corregidas de forma manual.
- Des-Rec (Desarrollo-Recurso): son la combinación de las dos anteriores, contienen dos campos, uno para introducir una respuesta teórica y otro para indicar un recurso. También deben corregirse de forma manual.

Los hipervídeos se utilizarán para extender la capacidad de aprendizaje por parte del alumno, se tratan de vídeos enriquecidos que contienen información adicional al simple vídeo. Tiene las siguientes características:

- Indexado de partes de vídeos.
- Selección de subtítulos: permitiendo crear un contenido que pueda ser consumido por alumnos que hablen distintos idiomas.
- Información contextual sincronizada con el vídeo: consiste en información adicional al vídeo que está sincronizada para enriquecer el vídeo. Son imágenes, notas, mapas, etc.
- **Enlace a actividades y recursos: permitiendo el enlace a exámenes creados.**

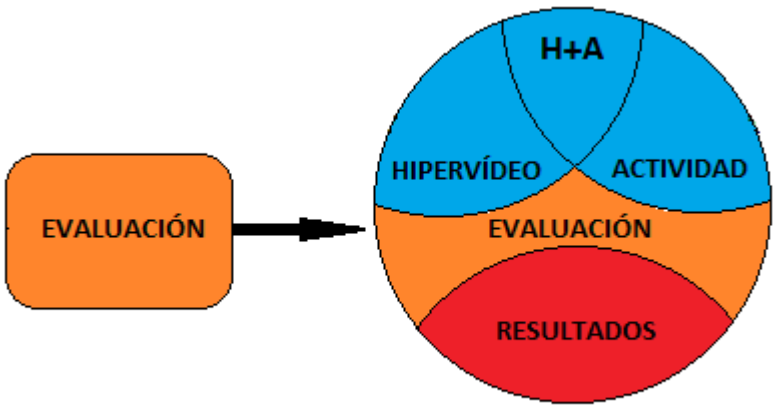


Ilustración 4: Evaluación

La **evaluación consiste en la selección, publicación y despliegue de una aplicación web que contiene los hipervídeos y actividades. Se encargará de**

permitir visualizar el hipervídeo y realizar las actividades con la posterior evaluación de las actividades. Por último se realiza la acción de guardar los resultados y respuestas de cada alumno. La aplicación web corregirá y puntuará las preguntas de las actividades de tipo test de forma automática pero los otros tipos de preguntas deben ser corregidas manualmente por el profesor a través del sistema.

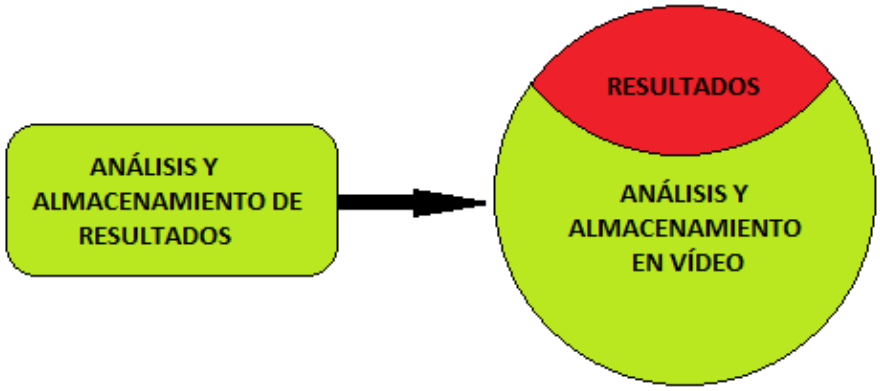


Ilustración 5: Análisis y almacenamiento en vídeo

El **análisis puede ser realizado a partir de los resultados usando servicios externos como Flubaroo**, ya que la aplicación ha sido diseñada para permitir la compatibilidad con este servicio.

Con el almacenamiento de los resultados y respuestas de los alumnos, una vez corregidas, conseguimos tener una copia de seguridad de estos datos para posibles análisis futuros o recuperaciones futuras de estos ficheros. Para ello se usará un sistema de almacenamiento en vídeo.

Este sistema es escalable y puede ser implantado en cursos online masivos pudiendo mejorar la situación de este tipo de cursos. Permite el almacenamiento de datos masivos mediante el uso de vídeos como base de datos de historiales.

Por este motivo, aunque este sistema puede ser usado en distintos tipos de educación online, se centra en los MOOCs dado que combate el problema del abandono de estos cursos y permite el almacenamiento de datos masivos de una forma segura, pudiendo ser usados esos datos en el análisis de datos para la obtención de estadísticas y sacar conclusiones que mejoren la educación por parte del profesor.

Capítulo 2.- VideoData ante el problema de los MOOCs

En este capítulo se abordan los MOOCs debido al auge que han tenido y al potencial que ofrecen para llevar el conocimiento a cualquier parte del mundo. Aunque se comentará y definirán los MOOCs, la parte más interesante serán los problemas que tienen para explotar todas las posibilidades que ofrece.

Existen bastantes investigaciones sobre los MOOCs, comentando estos en términos generales, resaltando algunas características, evaluando como se están utilizando, analizando los problemas existentes, etc. Por este motivo este capítulo no es una investigación en profundidad sino el análisis de trabajos existentes.

¿Qué es un MOOC?

"Los Cursos Masivos Abiertos en Red, o MOOC (del inglés, Massive Open Online Courses), son una nueva modalidad de formación con propuestas orientadas a la difusión web de contenidos y un plan de actividades de aprendizaje abierto a la colaboración y la participación masiva. Son cursos con soporte web escalable e inscripción libre para quienes quieran acceder y seguir la propuesta formativa."[2].

Características de un MOOC

Las características las indican sus siglas pero son conceptos abiertos a distintas interpretaciones [3]. Algunas de ellas son:

- *Abierto*: va más allá acerca de la gratuidad del cursos, puede referirse a que está disponible y es accesible a los usuarios.
- *Masivo*: están diseñados para ser masivos, va implícito en el concepto anterior ya que al ser accesible para un gran número de usuarios se consigue que sea masivo.
- *En linea*: son totalmente a distancia y se utilizan herramientas de comunicación social en la interacción del curso entre los alumnos.
- *Curso*: son concebidos con una estructura clara orientada a la realización de un conjunto de tareas en un periodo determinado. También deben tener unos objetivos docentes y debe realizase algún tipo de evaluación a los alumnos para comprobar si han aprendido los conocimientos propuestos.

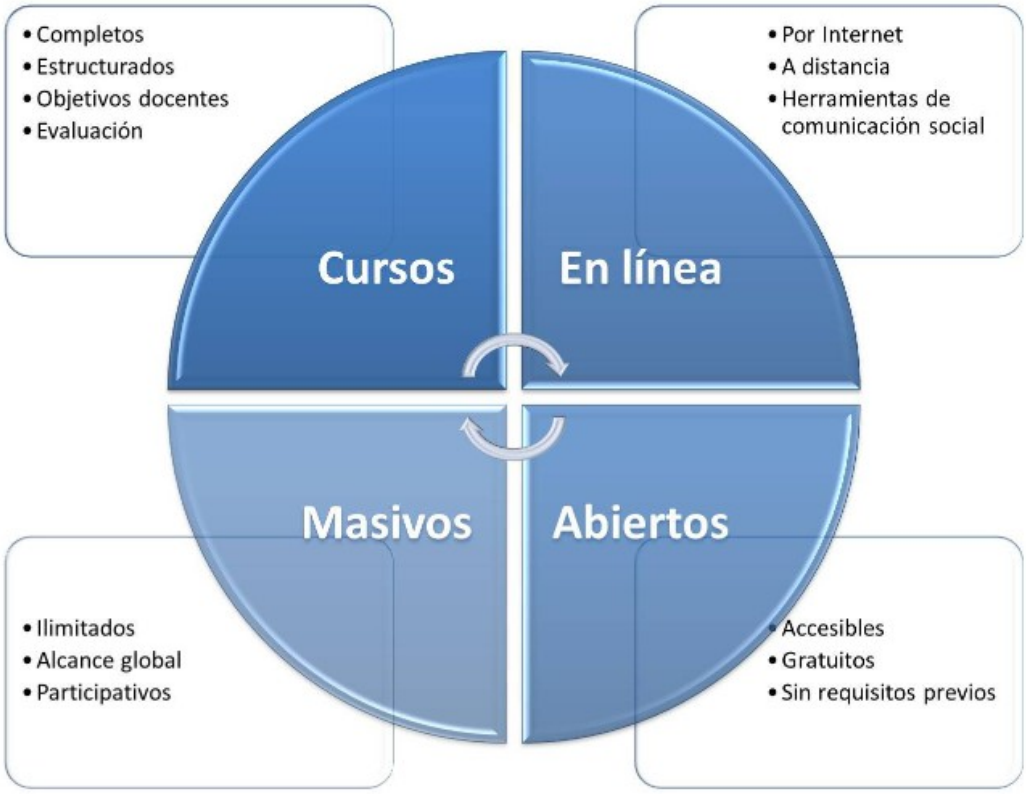


Ilustración 6: Características principales asociadas al concepto MOOC [3]

Para ser considerado MOOC deberían tener estas cuatro características principales, actualmente hay muchos cursos definidos como MOOC que en realidad no lo son a pesar de que las cuatro características pueden tener diferentes interpretaciones.

Por ejemplo en la definición de *curso* aunque abarca la evaluación de los alumnos, existen muchos MOOCs que no realizan evaluaciones del material o conceptos impartidos en dicho curso.

Tipos de MOOCs

Los MOOCs con contemplados desde diferentes puntos de vista en función de su estructura y diseño, modificando tanto la visión sobre el aprendizaje sobre como debe ser conseguido por parte de los alumnos. Aunque pueden distinguirse muchos tipos de MOOCs [4] nos centraremos en los dos tipos principales:

- cMooc: la 'c' significa *conectivista*, utiliza principalmente la interacción entre los alumnos compartiendo documentos, realizando trabajos en grupo, etc centrando el aprendizaje en el trabajo en equipo. Son muy abiertos en términos de personalización sobre cada alumno teniendo como consecuencia la dificultad de evaluación.
- xMooc: la 'x' proviene de las diferentes plataformas más conocidas

(Coursera, Miriada, Edx, Moodle, etc) y se centran en el enfoque de una estructura más restrictiva mediante una web que gestiona todas las facetas: contenido, usuarios, etc.

Aunque tienen una *filosofía* diferente, comparten las características de los MOOCs dado que son cursos masivos, están abiertos y por supuesto son online [5].

Análisis de los MOOCs

En sus inicios los MOOCs estaban en pleno auge debido a que permiten acceder a contenidos de niveles superiores desde tu propia cada. Un término muy significativo usado es *la democratización de la educación* [6] y hace referencia a poder acceder a material que antes solo era accesible desde la Universidad.

En pleno auge de los MOOCs se integraron en la educación superior y en las Universidades a través de distintas plataformas. Plataformas como **Miriada X** que fue impulsada por 18 universidades iberoamericanas con 58 cursos gratuitos en sus inicios. Otras como **Coursera** que es la que agrupa a más estudiantes y universidades ofreciendo más de 200 cursos en inglés entre 33 universidades y tiene más de 2 millones de usuarios [7].

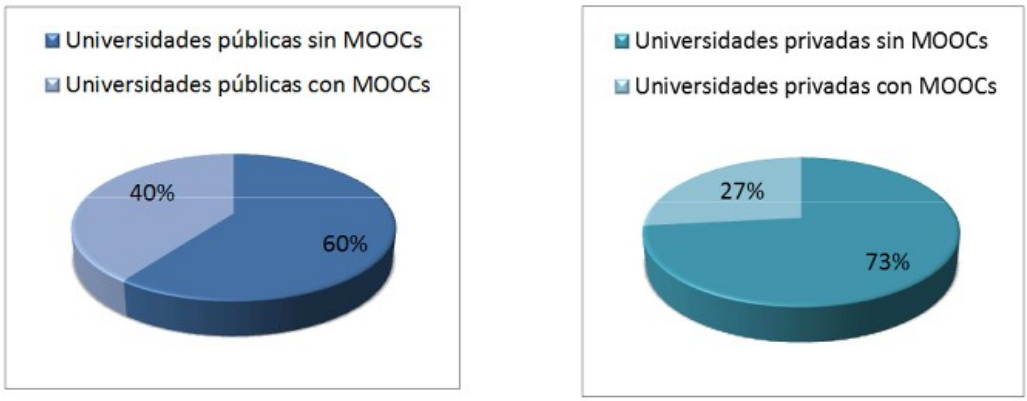


Ilustración 7: Comparativa sobre el uso de MOOCs en las universidades españolas [8]

Resulta muy interesante la comparativa distinguiendo entre universidades públicas y privadas en relación a la impartición de MOOCs.

En este estudio detallado [8] se realiza un gran trabajo investigando los MOOCs y su implantación por parte de las universidades, se comparan el uso de MOOCs en las universidades españolas con respecto a las universidades europeas obteniendo que ha tenido éxito mayoritariamente en España. También distinguen las áreas de conocimientos con mayor número de cursos de estos tipos destacando la ciencia y tecnología con el 22% de los MOOCs y las ciencias sociales con el 21%.

En términos económicos, ideados para que sean abiertos y masivos ha sido algo clave para explicar **el diseño escalable que permite la reutilización de contenido** reduciendo los costes de creación de cursos. Aunque en las plataformas se ofrecen cursos MOOCs gratuitos, también nos podemos encontrar otros cursos definidos como MOOC que es de pago y suele permitir obtener un certificado a los alumnos. Estos cursos de pago unido a la financiación por parte de gobiernos [9] y empresas junto al diseño escalable hace viables estas plataformas de MOOCs.

Aunque no hay definido un modelo de negocio estable han resistido bien la crisis económica ya que "surgen los MOOC como respuesta a la necesidad de acercamiento entre la formación y la sociedad del conocimiento. Pero que los cursos masivos y abiertos marquen el futuro de la educación dependerá en gran medida de su viabilidad económica." [10]. Y es que en tiempos de crisis la formación es un factor clave para obtener un trabajo de tal modo que el uso de MOOCs que son gratis o *baratos* es una oportunidad que no es desaprovechada, constituyendo una opción más económica que la educación superior y obteniendo una formación concreta que puede tener incluso certificación.

Algunos problema de los MOOCs

A pesar de la buena intención con la cual fueron creados y todo el potencial que ofrecen, no han terminado de despegar como se esperaba debido a varios motivos importantes, entre ellos:

- **Contenido de baja calidad:** deben tener una estructura y diseño adaptable y acorde.

	Muy baja		**Baja**		**Media**		**Alta**	
	n	%	*n*	%	*n*	%	*n*	%
Miríada X	-	-	5	20	14	56	16	24
Coursera	1	4.2	6	25	14	58.3	3	12.5
Udacity	-	-	-	-	8	**80**	2	**20**
Unimooc	-	-	3	25	6	50	3	25
Openlearning	-	-	2	33.3	4	**66.7**	-	-
OpenHPI	-	-	-	-	5	**71.4**	2	28.6
UNED COMA	-	-	4	18.2	17	**77.3**	1	4.5
Class2Go	-	-	1	12.5	7	**87.5**	-	-
Canvas Network	-	-	2	25	6	75	-	-
EdX	-	-	1	14.3	6	**85.7**	-	-

Ilustración 8: Porcentajes de la calidad pedagógica de las plataformas MOOC [11]

Sin embargo, muchos MOOCs no son diseñados por educadores obteniendo en vez de un curso un conglomerado de material sin formato ni sentido ninguno [11].

- **Alta tasa de abandono:** alrededor del 90% de los estudiantes suscritos a algún MOOC no lo termina. Puede ser debido a un bajo compromiso de trabajo para terminar con éxito un MOOC [12]. Aunque las causas son más numerosas como el citado anteriormente ***contenido de baja calidad,*** **la ausencia en algunos casos de certificación debido a problemas de evaluación masivas**, metodologías rutinarias y aburridas, etc [13].

- **Son comerciales:** cada vez más plataformas anuncian sus MOOC como un producto más comercial teniendo que pagar altos pagos por el curso y perdiendo su definición como MOOC debido a que deja de ser masivo [14] [15].

Uno de los mayores errores fue tratar a los MOOC con unas expectativas tan altas, ahora ante los grandes problemas que tiene debe mutar para poder sobrevivir. Se puede consultar como han tenido que ser modificado *su estructura* con el objetivo de no desaparecer en este interesante artículo [16].

Propuesta del sistema VideoData como solución

Con este sistema no se pretenden resolver todos los problemas sino mejorar la situación de algunos inconvenientes que tienen los MOOC. **El sistema se centra en mejorar la creación de contenido de los MOOC proponiendo el uso de hipervídeos para crear un contenido más valioso y a la vez mejorará la tasa de abandono debido a que el contenido de baja calidad y aburrido son unos factores importantes en el abandono de los cursos.**

"Si llevamos a cabo la evaluación de los hipervídeos didácticos considerando sus implicaciones en el aprendizaje de los alumnos, habría que tener en cuenta sus efectos en la atención, motivación, comprensión, etc." [17]

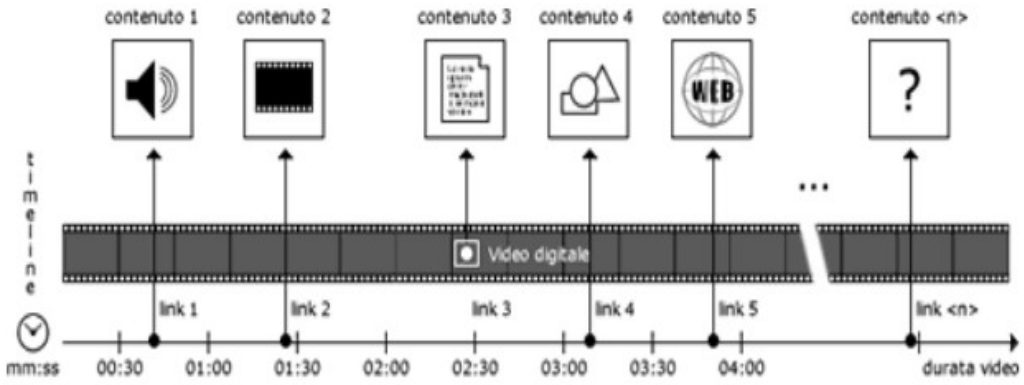


Ilustración 9: Estructura de un hipervídeo [17]

"El hipervídeo es un modelo de vídeo interactivo basado en la asociación de contenidos de diversa naturaleza a lo largo de su línea narrativa. Se trata de un hipertexto audiovisual, de manera que se puede intervenir en la secuencialidad del relato e interactuar con otros tipos de información: textos, imágenes fijas, audio, páginas web, etc." [17]

Además del empleo de hipervídeos como material de aprendizaje, **se propone el uso de actividades para realizar evaluaciones**, por ejemplo pequeños formularios con un conjunto de preguntas que interaccionan con el hipervídeo y están integrados en su estructura. De esta forma los alumnos estarán poniendo a prueba los conocimientos y el curso será tomado más en serio bajando el porcentaje de abandono.

Un factor clave es la calidad del contenido por esto el profesor creador del contenido debe tener en cuenta que el hipervídeo consigue mejorar la formación sobre los alumnos pero el contenido debe ser de calidad. Por ello en la siguiente tabla se indican los conceptos a tener en cuenta para conseguir un contenido que favorezca el éxito en los cursos MOOC.

1. Calidad técnica y estética	
1.1 Presentación de la información	• Elementos de información variados (audio, imágenes estáticas, imágenes en movimiento…) • Tamaño de los textos y gráficos proporcionales. • Imágenes nítidas que facilitan la comprensión. • La imagen y el sonido se complementan. • La banda sonora se reproduce correctamente. • El vídeo utiliza con eficacia los recursos propios del lenguaje audiovisual.
1.2. Aspectos funcionales	• La navegación se realiza con facilidad. • El uso y manejo por parte del usuario es simple y sencillo. • Presenta múltiples vínculo o enlaces. • La estructura es clara, sencilla e intuitiva. • La navegación se adapta a las respuestas y necesidades de los usuarios. • El usuario controla el ritmo de interacción y decide cuando activar un vínculo o volver al vídeo conductor
2. Aspectos didácticos	
2.1. Objetivos	• Los objetivos del material didáctico son pertinentes. • La organización de los contenidos en el recurso facilita el logro de los objetivos de aprendizaje propuestos.

2.2. Contenidos	• La información que se presenta está actualizada, es pertinente y relevante. • El contenido está organizado correctamente. • La información se presenta en forma clara y precisa. • El volumen de información que se proporciona es adecuado según el contenido abordado. • El ritmo de la presentación de la información es adecuada respecto al tema y a la audiencia.
2.3. Actividades	• Facilita la realización de diversas tareas. • Fomenta la realización de ejercicios posteriores. • Proporciona elementos para la discusión y el debate.
2.4. Estructura del mensaje	• El medio reemplaza ventajosamente a otro mensaje de concepción tradicional. • El medio se adecua al contenido abordado. • La duración del vídeo es pertinente con la audiencia y con el contenido abordado. • El medio invita al empleo de materiales complementarios. • Los recursos que aportan los enlaces apoyan la comprensión del mensaje del vídeo conductor.
2.5. Evaluación	• El recurso ofrece algún modelo o instrumento de evaluación de los aprendizajes. • El tipo de evaluación se relaciona explícitamente con los objetivos y contenidos planteados.
2.6. Alumnos	• Estimula la participación del alumno. • Presenta elementos motivadores. • Mantiene la atención del alumno. • Estimula la imaginación y creatividad. • Promueve la activación de diferentes operaciones cognitivas. • Fomenta la iniciativa y la toma de decisiones • La estructura hipertextual del recurso favorece los aprendizajes • Promueve el autoaprendizaje o aprendizaje autónomo. • Posibilita el trabajo colaborativo.
2.7. Profesor	• Permite la participación del profesor para adaptar el documento a distintas situaciones curriculares. • Complementa la información proporcionada por el profesor para mejorar el proceso de enseñanza.

2.8. Guía Didáctica	• El recurso se acompaña de una guía didáctica que contempla los objetivos y las características del mismo. • La guía contempla bibliografía y recursos de referencia sobre el contenido que se aborda. • La guía contiene sugerencias didácticas y ejemplos de utilización para su integración curricular. • La guía contiene actividades complementarias.

Tabla 1: Modelo para el diseño y análisis de hipervídeos [17]

Cada punto de esta tabla puede ser cumplida gracias a la combinación de los hipervídeos y las actividades o exámenes. Con los exámenes se puede conseguir la evaluación y con la combinación de ambos elementos se mejora de forma radical el contenido más tradicional.

Además **debido a la masividad de estos cursos se desarrollará una forma de almacenamiento poco convencional, utilizando vídeos como almacenamiento de los historiales y resultados de las evaluaciones de los exámenes**. **Simulando** en cierta forma el viejo almacenamiento en **cintas magnéticas pero con la ventaja de poder acceder a un fichero mediante acceso aleatorio** a diferencia del acceso secuencial en las viejas cintas.

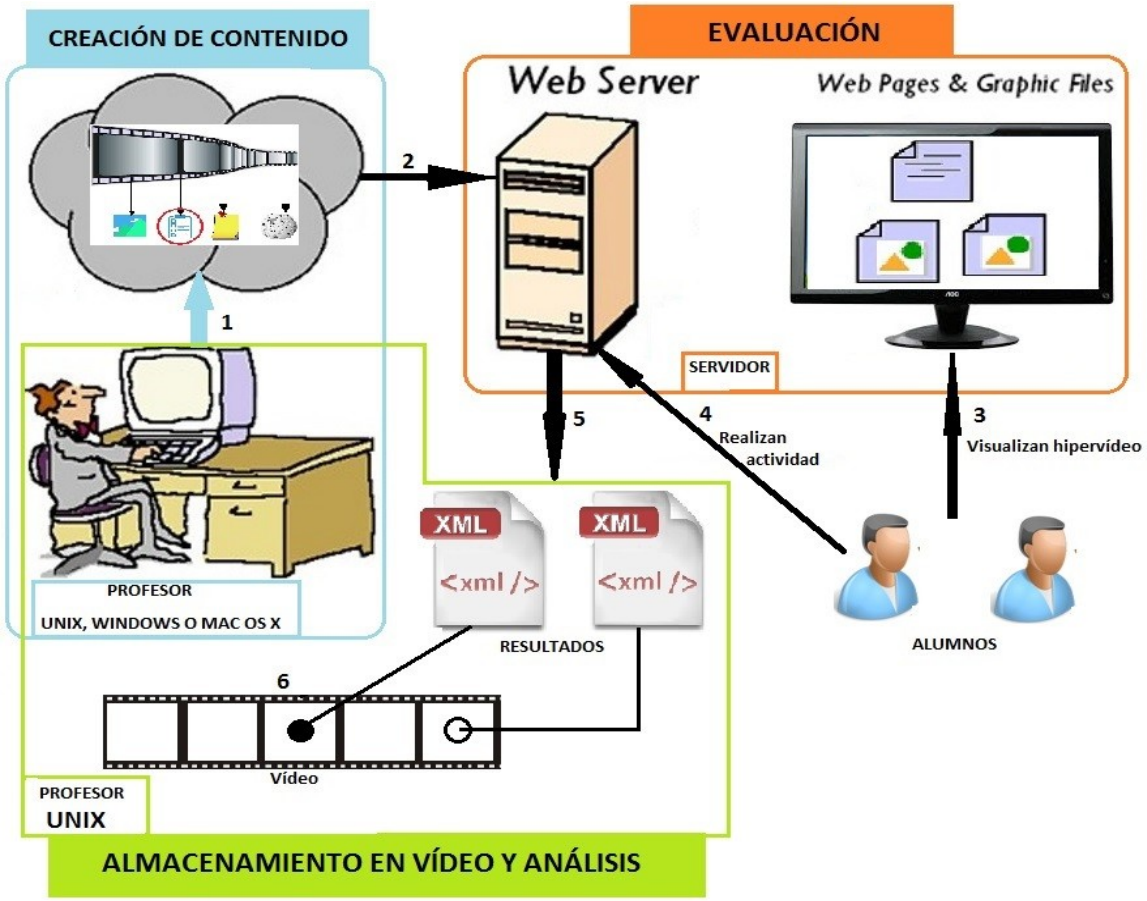


Ilustración 10: Arquitectura del sistema VideoData

En la imagen anterior se puede apreciar la arquitectura del sistema que se desarrollará. Se divide en tres módulos:

- **Creación de contenido:** el profesor creará los hipervídeos y las actividades (exámenes) pudiendo integrar en los hipervídeos los exámenes. **La creación de exámenes es compatible con el servicio externo Flubaroo** [1].
- **Evaluación:** una vez creado el contenido, se seleccionan las actividades e hipervídeos que se usarán, se publican y se despliega la aplicación web que los contiene. De esta forma los alumnos podrán visualizar los hipervídeos y realizar las actividades. Por último las actividades serán corregidas automáticamente (*las preguntas de tipo test*) y manualmente si hay otros tipos de preguntas.
- **Análisis y almacenamiento:** los resultados son compatibles con Flubaroo permitiendo el análisis de estos resultados por parte del servicio externo. Finalmente los resultados y respuestas son almacenados en vídeo, guardándose los datos para futuras consultas del historial.

Situación técnica

Desarrollo de software dirigido por modelos (MDD)

Popcornjs

Aplicaciones web J2EE

OpenCV en el manejo de vídeos e imágenes

Capítulo 3.- Análisis y diseño

En esta parte se procederá a realizar el análisis y diseño de la aplicación. Es una parte imprescindible en cualquier aplicación de un tamaño aceptable debido a que permite entender de una forma fácil el sistema y si en algún momento este sistema es usado por un desarrollador podrá ser modificado fácilmente para extenderlo y lograr sus propósitos o incluso pueden ser utilizados los módulos que le interese.

Esquema del sistema y tecnologías usadas

En este apartado se comentará el funcionamiento íntegro del sistema junto a las tecnologías que son usadas en cada módulo. Se procederá a realizar un análisis de cual es el motivo del uso de cada tecnología.

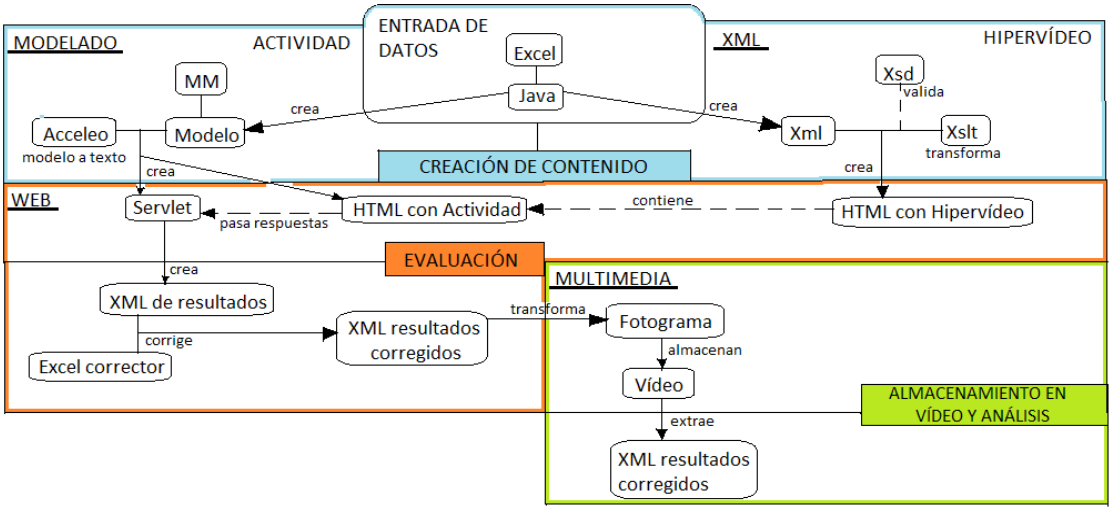


Ilustración 11: Esquema del sistema y tecnologías usadas

La imagen anterior contiene todo el funcionamiento del sistema unificado de una forma conceptual junto a las tecnologías que son usadas para cada tarea. Además están separados cada parte del sistema por módulo, diferenciando entre la creación del contenido, la evaluación y el almacenamiento en vídeo. Cada módulo usa distintas tecnologías, concretamente:

- **Creación de contenido**: se pueden diferenciar tres apartados:
    - Entrada de datos: consiste en la entrada de datos de las hojas de cálculo y su transformación en otros tipos de ficheros usando java.
    - **Modelado**: para trabajar con las actividades se utiliza el modelado de software. Hay un metamodelo diseñado que contiene la sintaxis, estructura y restricciones de **las actividades que serán los modelos creados desde la entrada de datos**. Además una vez creado las

actividades o modelos, se procederá a la transformación mediante **Acceleo** que es un lenguaje de modelo a texto (m2t) que transformará el modelo en código.

- **XML:** el manejo de los hipervídeos se realiza mediante tecnologías xml, en una primera fase se creará por cada hipervídeo un fichero xml que debe ser validado contra un fichero **xsd**. Una vez validado, se procederá a la transformación del xml usando **xslt** para transformar estos ficheros en html que contendrán la visualización del hipervídeo desde un navegador.

- **Evaluación:** en la evaluación se utilizan dos fases, consiste en la visualización de los hipervídeos y la realización de las actividades, contiene las siguientes tecnologías:

  - **WEB:** concretamente es una aplicación J2EE con el contenido creado. De esta aplicación web forman parte los hipervídeos mediante un fichero html, y las actividades que contienen un fichero html que es la visualización de la actividad a realizar por los alumnos y un servlet que se ejecuta en el servidor con la función de evaluar las preguntas de la actividad y guardar los resultados.

  - **Java:** debido a que las actividades tienen algunos tipos de preguntas que no pueden ser corregidos de forma automática, se ha implementado usando java un sistema que se encarga de puntuar las preguntas no corregidas automáticamente de cada alumno. Consiste en rellenar los valores a cada pregunta y cada alumnos desde una hoja de cálculo y a partir de ahí se actualizarán los resultados de los alumnos.

- **Almacenamiento en vídeo y análisis:** este módulo es el más novedoso y se realiza algo que es insólito como es el guardar información textual en un vídeo, en este caso los resultados y respuestas de los alumnos, y usar este vídeo como una base de datos. Se utiliza lo siguiente:

  - **Multimedia:** para el almacenamiento en vídeo concretamente se utiliza OpenCV para la creación y recuperación de los resultados y

respuestas de los alumnos. El primer paso al crear el vídeo es guardar estos ficheros de respuestas de los alumnos en fotogramas y después crear el vídeo usando todos los fotogramas además de crear un índice para poder obtener ficheros de resultados de una forma eficiente. Además de crear el vídeo, lógicamente también se permitirá la recuperación de determinados ficheros o todos a partir del vídeo.

- **XML:** para el análisis de resultados hay que utilizar herramientas de terceros, la potencia es increíble ya que se guardan muchos datos de cada alumno (resultados por tipo de pregunta, respuestas concretas, etc) en formato XML que es un lenguaje interoperable que permite trabajar de forma simple para poder obtener estadísticas y proceder al análisis de estos datos.

Casos de uso

En este apartado se procederá con la exposición de los casos de usos (CU) que consistirán en los funcionalidades principales que podrá realizar el usuario.

Actores

Es sistema será utilizado por los siguientes actores:

- **Profesor:** será el usuario principal del sistema ya que es el que utiliza la mayoría de funcionalidades del sistema y el que saca todo el provecho de la aplicación.
- **Alumno:** es un actor secundario pero imprescindible ya que el objetivo de la aplicación es mejorar el sistema de aprendizaje online, por lo tanto será usado todo el contenido creado por los alumnos.

Diagramas de casos de uso

El objetivo de los casos de uso es describir las actividades que podrán utilizarse en el sistema en relación a cada actor. De esta forma queda delimitado tanto la funcionalidad como quien utiliza cada función.

Los casos de usos que se mostrarán a continuación han sido separados por actor y

por módulo del sistema. Estos serán comentados de forma breve ya que se ampliará cada funcionalidad en los posteriores diagramas (*de clase y de secuencia*).

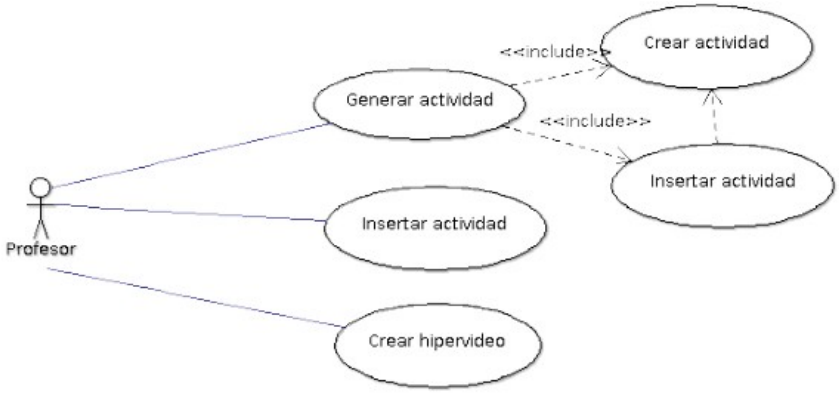


Ilustración 12: CU (Creación de contenido)

El caso de uso anterior describe el módulo de creación de contenido y las opciones que se ofrecen. El actor encargado de estas tareas el el profesor y dichas tareas serán la primera fase a realizar por este actor.

Una vez realizadas estas tareas, ya estarán disponibles las actividades que hayan sido creadas e insertadas y podrán ser utilizadas por parte de los alumnos, y lo mismo para el hipervídeo.

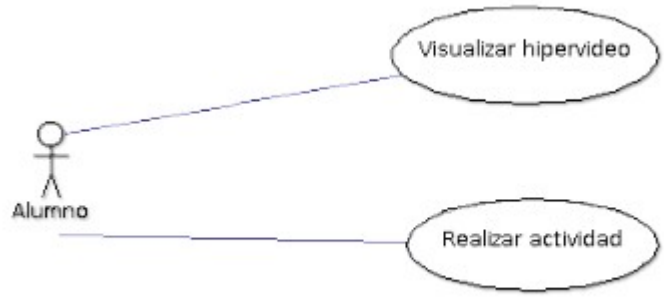


Ilustración 13: CU (del actor Alumno)

Los casos de usos de los alumnos consisten en utilizar el contenido creado por parte del Profesor. Concretamente es visualizar el hipervídeo interaccionando con todos los eventos que contiene y realizar actividades que consistirá en rellenar un

formulario contestando un conjunto de preguntas.

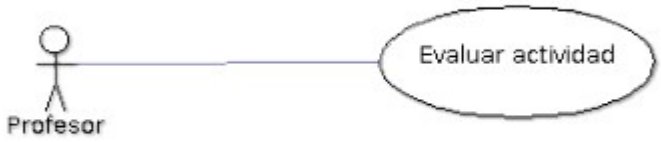


Ilustración 14: CU (Evaluación)

El CU anterior muestra la evaluación de la actividad por parte del profesor una vez que los alumnos han realizado dicha actividad. Esto puede consistir en ver simplemente las respuestas de cada alumno o evaluar cada pregunta que no es corregida de forma automática.

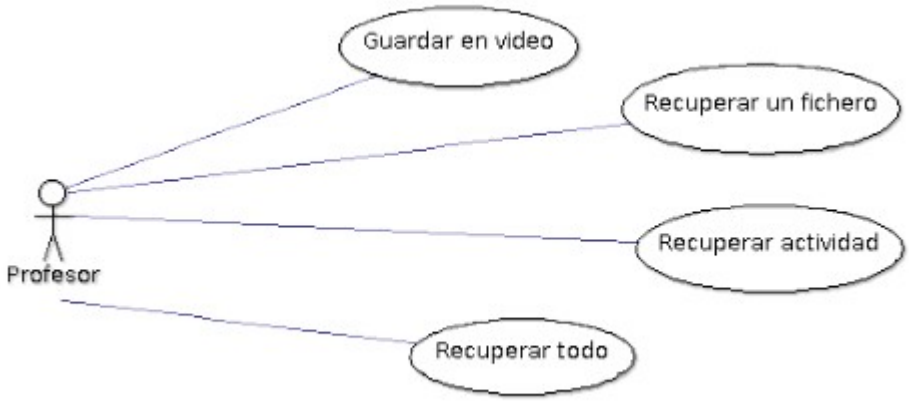


Ilustración 15: CU (VideoData)

La imagen anterior muestra todas las funcionalidades relacionadas con el almacenamiento de los resultados de los alumnos en vídeo y la posterior recuperación de estos ficheros a partir del vídeo.

Modelo conceptual

En este apartado se realizará el modelo conceptual sin entrar en detalles y se comentará como funciona el sistema y cada una de sus funcionalidades. Para ello se ha separado en dos módulos que están claramente diferenciados:

- **Creación de contenido y evaluación**.
- **VideoData**: almacenamiento en vídeo y recuperación de la información.

Creación de contenido y evaluación

Esta sección incluye un diagrama conceptual que servirá como primera toma de contacto con el diseño de cada componente y se comentará como interactúa cada elemento con el resto.

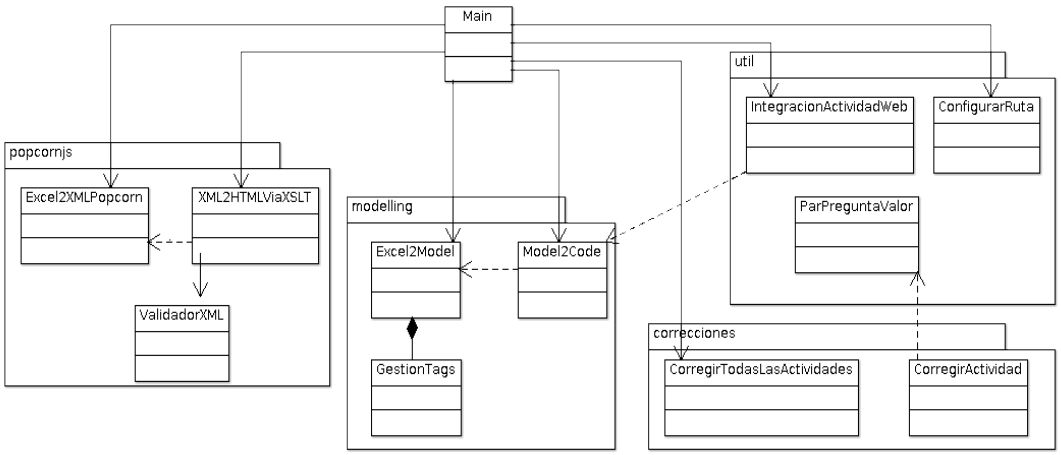


Ilustración 16: Diagrama conceptual (Creación y evaluación)

En la ilustración anterior se pueden ver los componentes que se utilizarán en el sistema de creación de contenido y evaluación de actividades. El sistema estará centralizado por la clase Main que será desde la que se ejecute la aplicación en formato de consola y que permitirá las opciones descritas.

La estructura de este sistema como se puede ver está organizada en paquetes por tipo de funcionalidad diferente. No se entrará ahora en detalle porque cada clase se describirá de forma más detallada en los diagramas de clase.

VideoData

Este apartado contiene el diagrama conceptual del paquete VideoData que se encarga de guardar la información textual (*los resultados de las actividades*) en vídeo y la posterior recuperación de la información usando este vídeo como base de datos.

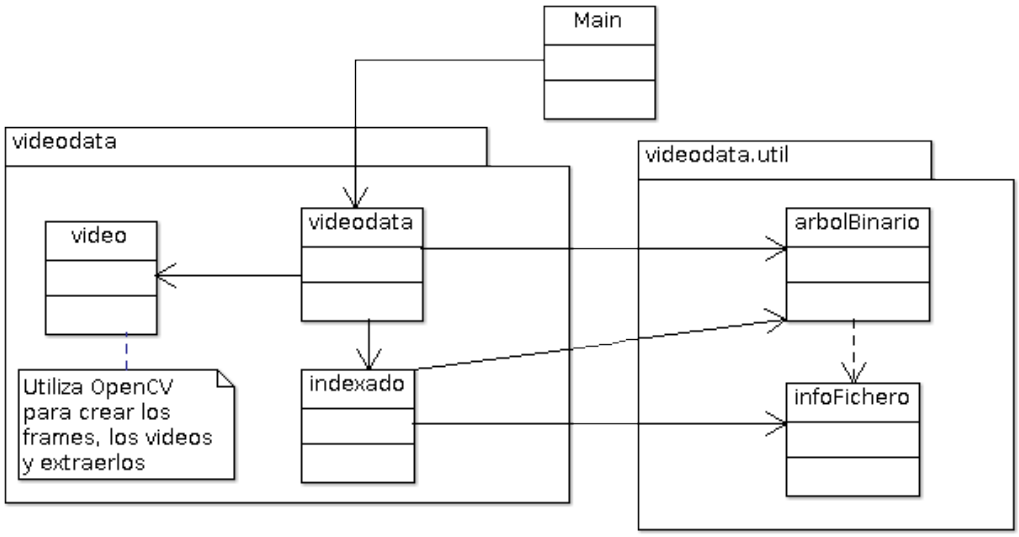


Ilustración 17: Diagrama conceptual (VideoData)

Este es el módulo encargado del tratamiento de la información a través del vídeo, permitiendo guardar los datos en el vídeo y realizar recuperaciones de la información a partir del vídeo usado como base de datos.

Para trabajar con el vídeo será necesario el uso de OpenCV limitando este módulo su implementación en C++ o Python que son los únicos que dan soporte total a OpenCV.

El lenguaje usado para desarrollar este módulo será Python debido a que es más simple que C++ y evita la necesidad de compilación del software. Al tratarse Python de un lenguaje de interprete nos ahorramos la compilación, además de ser más legible que C++.

El sistema VideoData será comentado de una forma más detallada en los posteriores diagramas.

Diagramas de clase

En este apartado se realizarán los diagramas de clase y serán comentados con detalle para tener claro para que sirve cada clase y elemento del sistema.

Como en los apartados anteriores se diferenciará entre dos módulos distintos, la creación y evaluación por una parte y VideoData por otra.

Creación de contenido y evaluación

Al tratarse de un diagrama de clases bastante grandes, se comentará cada paquete de forma separada para que sean diagramas legibles por el desarrollador.

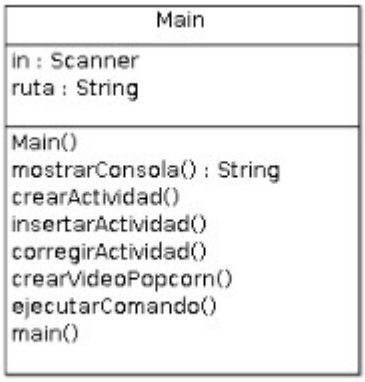


Ilustración 18: Clase Main

Esta es la clase principal del sistema de creación y evaluación. Desde el método *main()* se ejecuta en formato consola y permite seleccionar las funciones indicadas que coinciden con los casos de uso de creación de contenido y evaluación.

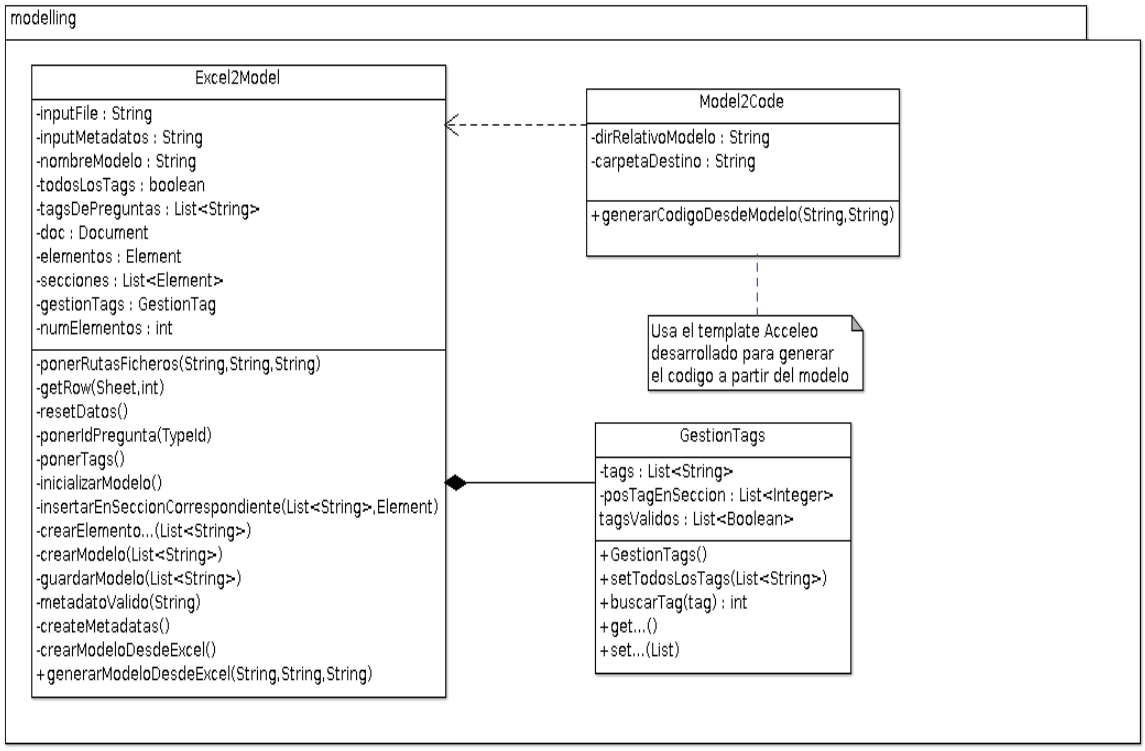


Ilustración 19: Diagrama de clase (modeling)

El paquete modeling se encarga de la creación e inserción de las actividades, para ello utiliza la clase **Excel2Model** para crear la actividad, como se indica en el propio nombre de la clase utiliza un fichero Excel para crear la actividad. Y esta clase utiliza la clase GestionTags para poder filtrar las preguntas que están en el Excel según el tag que tienen.

Una vez que está creada la actividad, la inserción de la actividad se realizará por la clase **Model2Code** que a partir de la actividad procederá a la generación del código necesario para poder utilizar la actividad. Para la generación de código se utilizará **Acceleo** debido a que las actividades serán modelos y se utilizará el metamodelado para trabajar con estas. En la sección de desarrollo correspondiente se comentará el metamodelo desarrollado y todo su entorno.

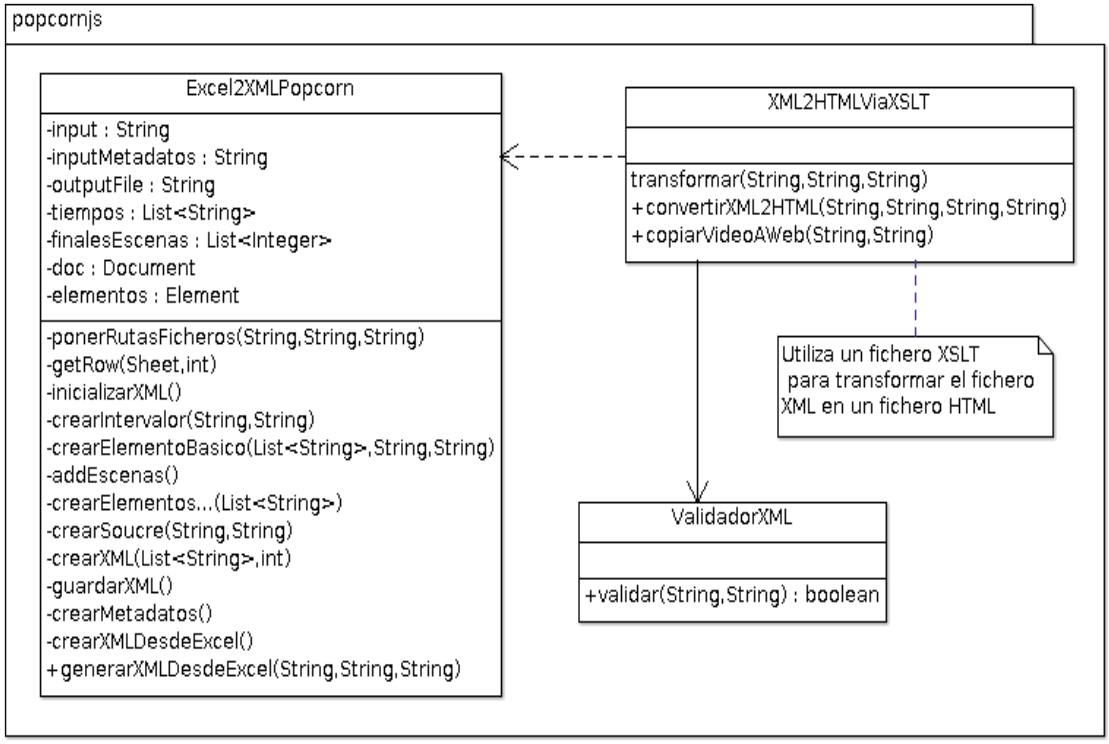


Ilustración 20: Diagrama de clase (popcornjs)

El paquete popcornjs se encarga de la creación de los hipervídeos, este proceso se realizará siguiendo una serie de instrucciones partiendo de la entrada de ficheros Excel para crear el hipervídeo y finalizando con un fichero html que permitirá la visualización del hipervídeo desde un navegador que lo soporte.

La clase **Excel2XMLPopcorn** se encarga de crear un fichero en formato xml que contiene todo lo necesario para crear el hipervídeo y será creado a partir de los ficheros Excel.

Una vez esté creado el fichero xml, la clase **XML2HTMLViaXLST** procederá primero a validar el xml usando la clase **ValidadorXML** que valida el xml generado usando un esquema (xsd) desarrollado. Y si el fichero xml es válido, se procede a realizar la transformación usando un fichero **XSLT** que transformará el xml en un fichero html con la estructura necesaria para la visualización del hipervídeo.

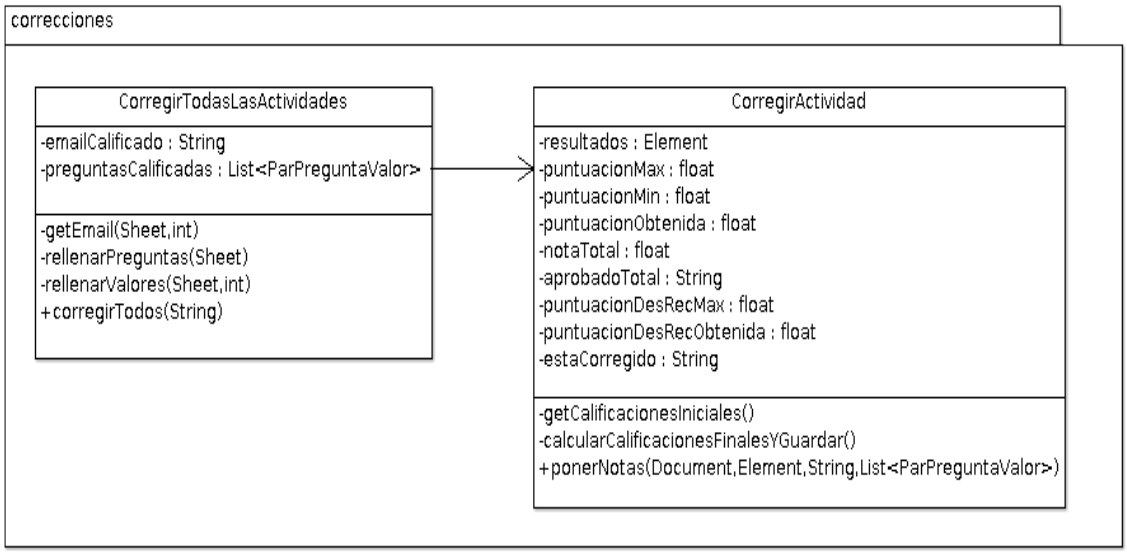


Ilustración 21: Diagrama de clase (correcciones)

El paquete de correcciones realiza la función de evaluación de los alumnos sobre las actividades realizadas por parte de estos. Concretamente lo realiza la clase **CorregirTodasLasActividades** que va leyendo de un fichero Excel las calificaciones que ha realizado el profesor sobre cada alumno y va corrigiendo la actividad alumno por alumno llamando al método **ponerNotas()** de la clase **CorregirActividad.** De esta forma se modifica el xml que es generado con las respuestas y resultados de cada alumno, actualizando los resultados con las calificaciones a las preguntas que no son corregidas de forma automática.

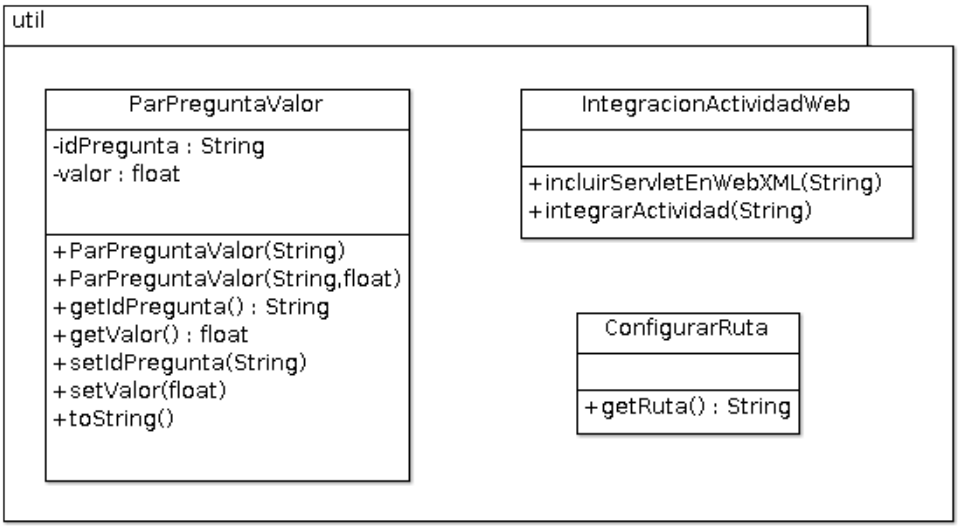


Ilustración 22: Diagrama de clase (util)

El paquete util contiene las utilidades que son usadas por otras clases, la clase **ConfigurarRuta** es utilizada para obtener la ruta absoluta donde se encuentra el proyecto, ya que se permitirá poner el proyecto en cualquier ruta y configurar el fichero **configuracionRuta.txt**.

La clase **ParPreguntaValor** es utilizado como POJO como clase contenedora para contener la información a tratar de una forma uniforme y encapsulada. Será utilizada en la evaluación ya que se usará una lista de estos elementos por alumno que contendrá los identificadores de las preguntas a evaluar y la puntuación o valor obtenido por parte del alumno.

La clase **IntegracionActividadWeb** es usada en el CU insertar actividad, ya que después de crear la actividad y el código fuente, hay que insertar los ficheros generados en la aplicación web VideoDataWeb.

VideoData (almacenamiento en vídeo)

En este apartado se mostrará y se comentará el diagrama de clases del módulo VideoData que se encarga del almacenamiento de información en vídeo y la posterior extracción de información del vídeo, usando los vídeos como base de datos.

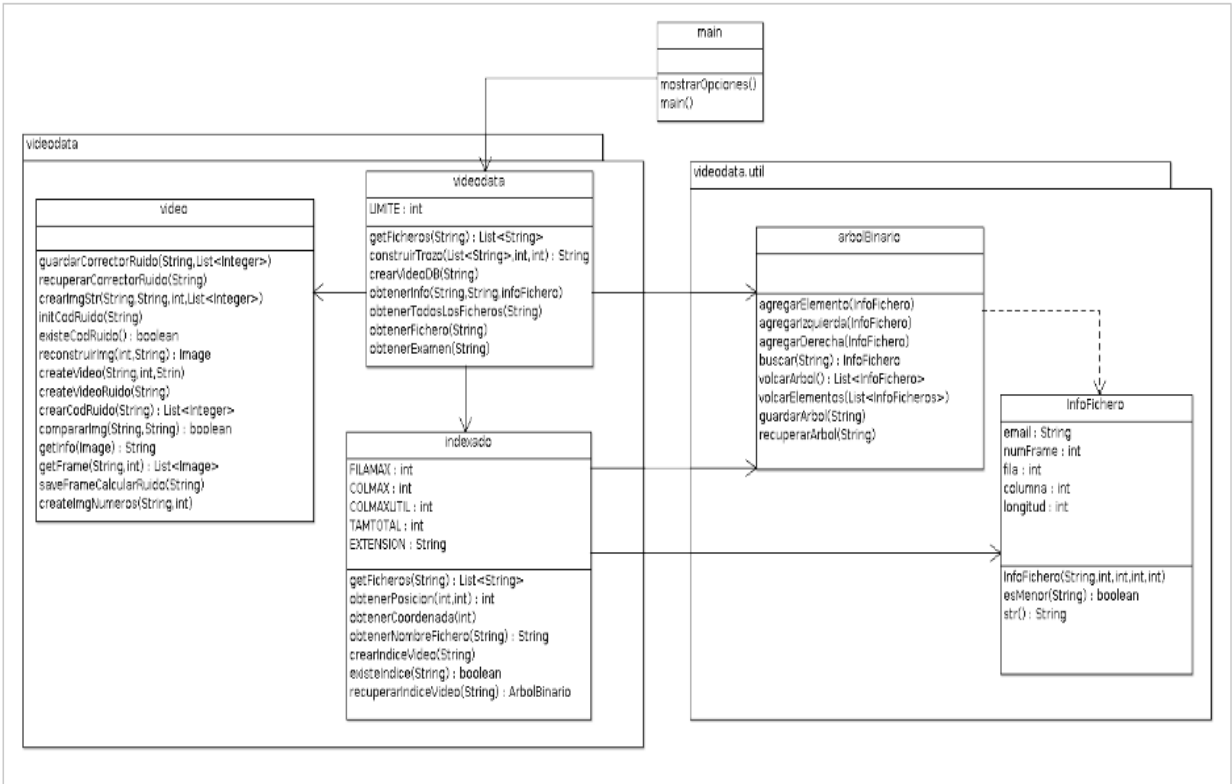


Ilustración 23: Diagrama de clase VideoData

La ilustración anterior contiene el diagrama de clase que se utilizará como módulo de VideoData. En el desarrollo de este módulo se usará Python ya que es necesario utilizar OpenCV.

La clase principal es **main**, desde aquí se llamará y se usará es resto de este sistema. Al ejecutar main, se mostrará una consola que permitirá el uso de instrucciones que coinciden con el CU VideoData.

Pero el fichero principal será **videodata** ya que contendrá los métodos principales y es el que interactúa con el resto de elementos. Desde el fichero **video** que es el encargado de crear y acceder a los vídeos usando OpenCV al fichero **indexado** que

se encarga de crear y recuperar el índice del vídeo para ofrecer un acceso rápido y eficiente a la información guardada en el vídeo. El indexado se comentará en el desarrollo con más detalle.

Pero básicamente utiliza la estructura de **arbolBinario** donde cada nodo será un fichero dentro del vídeo y la información que guarda cada nodo es la de la clase **InfoFichero** que contiene todo lo necesario para localizar el fichero de un alumno en el vídeo.

Diagramas de secuencia

Esta sección contiene los diagramas de secuencia para poder modelar la interacción entre los actores y los objetos del sistema. Se ha diseñado un diagrama de secuencia por cada caso de uso para describir la comunicación entre objetos en cada funcionalidad.

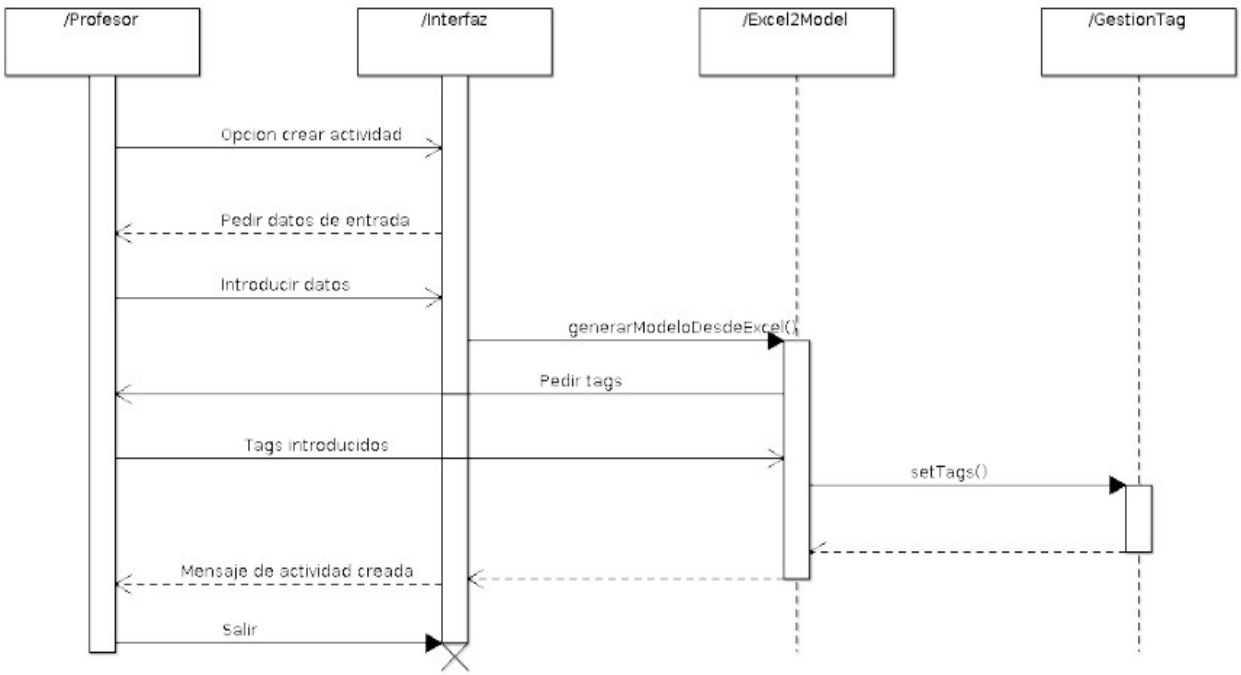


Ilustración 24: Diagrama de secuencia (Crear actividad)

La acción **Crear actividad** es realizada por el profesor a través de una interfaz que se trata de una consola, para ello se elige la opción y al pedir los datos se introducen los el nombre del fichero Excel de entrada y el nombre de la actividad. Después se

pide al usuario los tags a usar para filtrar las preguntas que compondrán la actividad y tras introducirse se crea la actividad informando al usuario de ello. Finalmente se procede a salir del sistema.

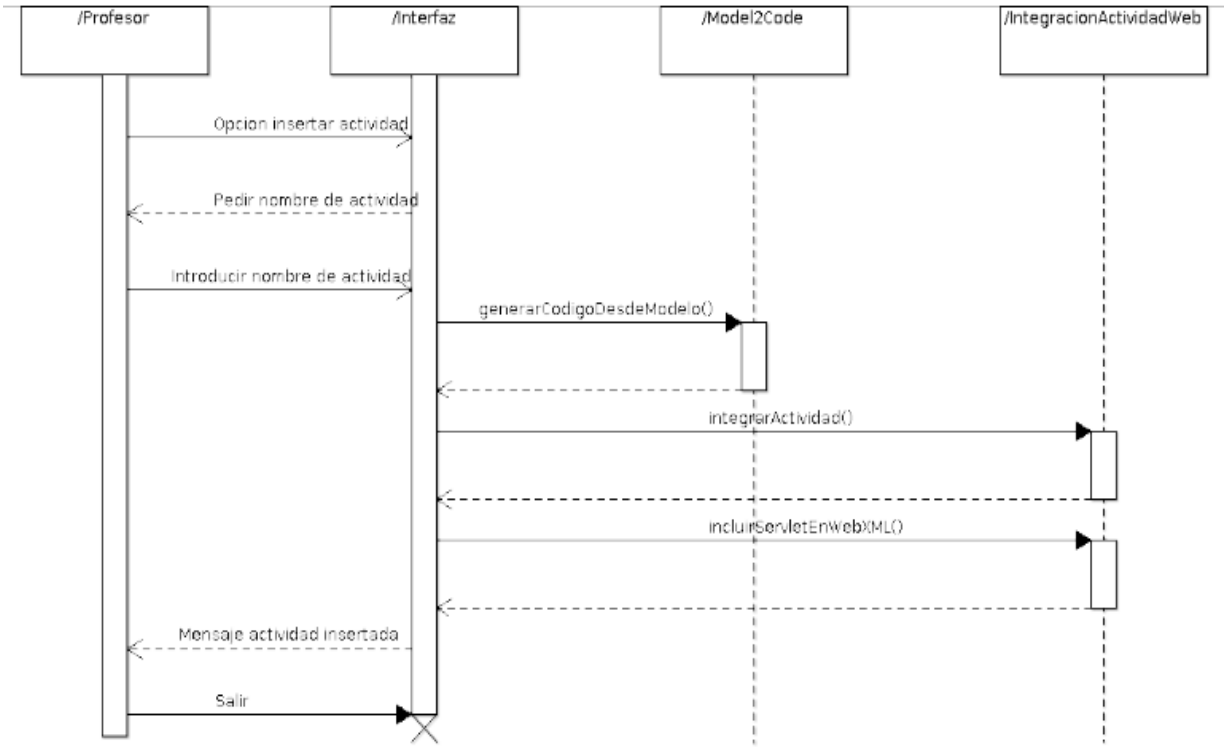


Ilustración 25: Diagrama de secuencia (Insertar actividad)

Para insertar una actividad ha tenido que haber sido creada con anterioridad. El proceso es muy simple, se elige esta opción y después se introduce el nombre de la actividad. Una vez hecho esto, se generará el código en la clase **Model2Code** y tras estos se copiará el código generado en la aplicación web VideoDataWeb y se declarará la actividad en el descriptor de despliegue de la aplicación web.

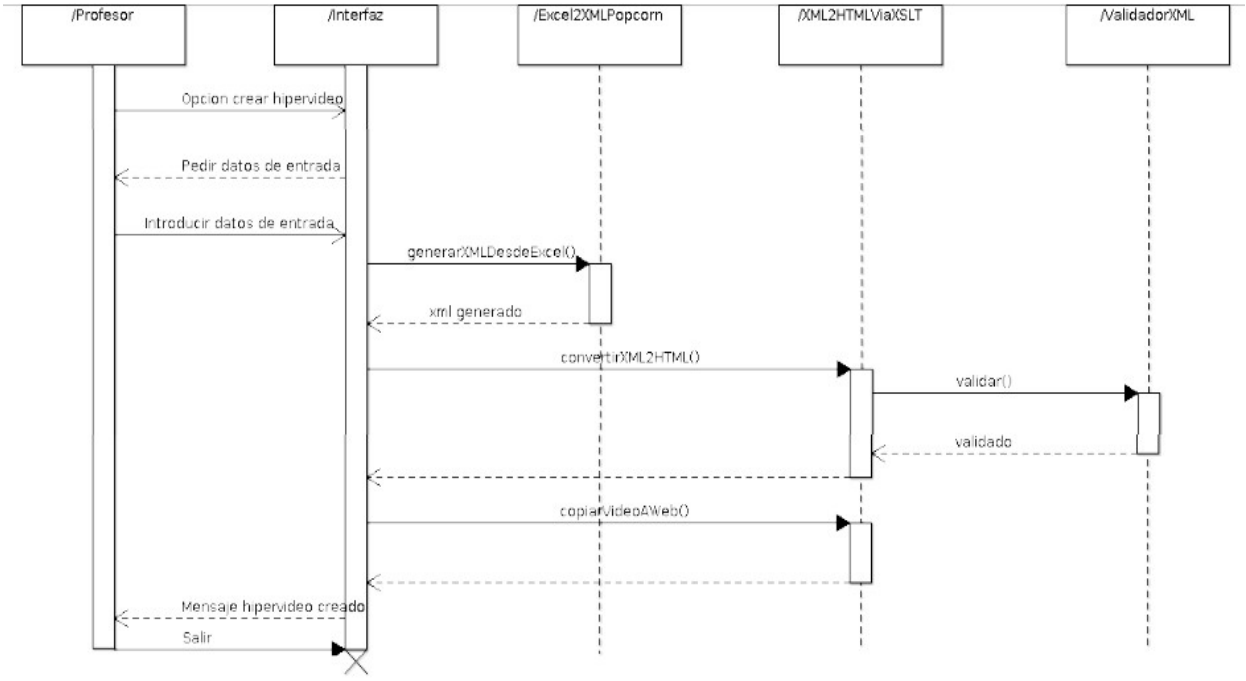


Ilustración 26: Diagrama de secuencia (Crear hipervídeo)

La creación de los hipervídeos sigue un proceso inicial similar ya que después de elegir esta opción, hay que introducir el nombre de los ficheros Excel de entrada y proporcionar un nombre al hipervídeo.

Después se creará a partir de los ficheros Excel un fichero xml, este será validado usando un esquema xsd desarrollado y se producirá a transformar el xml en un html que se utilizará para visualizar el hipervídeo usando un navegador, para ello usará un ficheros XSLT. De este proceso de transformación se encargará la clase **XML2HTMLVIAXLST.**

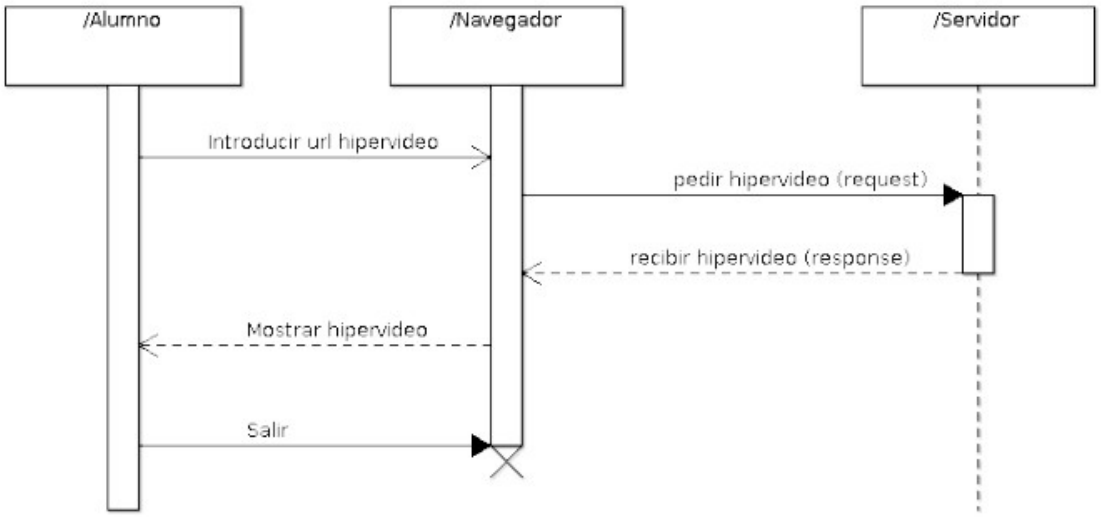


Ilustración 27: Diagrama de secuencia (Visualizar hipervídeo)

La visualización del hipervídeo será realizada por parte del alumno a través de un navegador que debe ser compatible con html5. El encargado de toda la tarea es el navegador que pedirá al servidor tanto el fichero html del hipervídeo que ha sido generado a partir de los ficheros de creación del hipervídeo, como todos los recursos que necesite este.

El alumno visualizará el hipervídeo interactuando con todos los elementos que contenga (*subtítulos*, *imágenes*, *wikipedia*, *mapas*, ***actividades***, *etc*).

Aunque lógicamente de la interacción con las actividades se encarga el sistema.

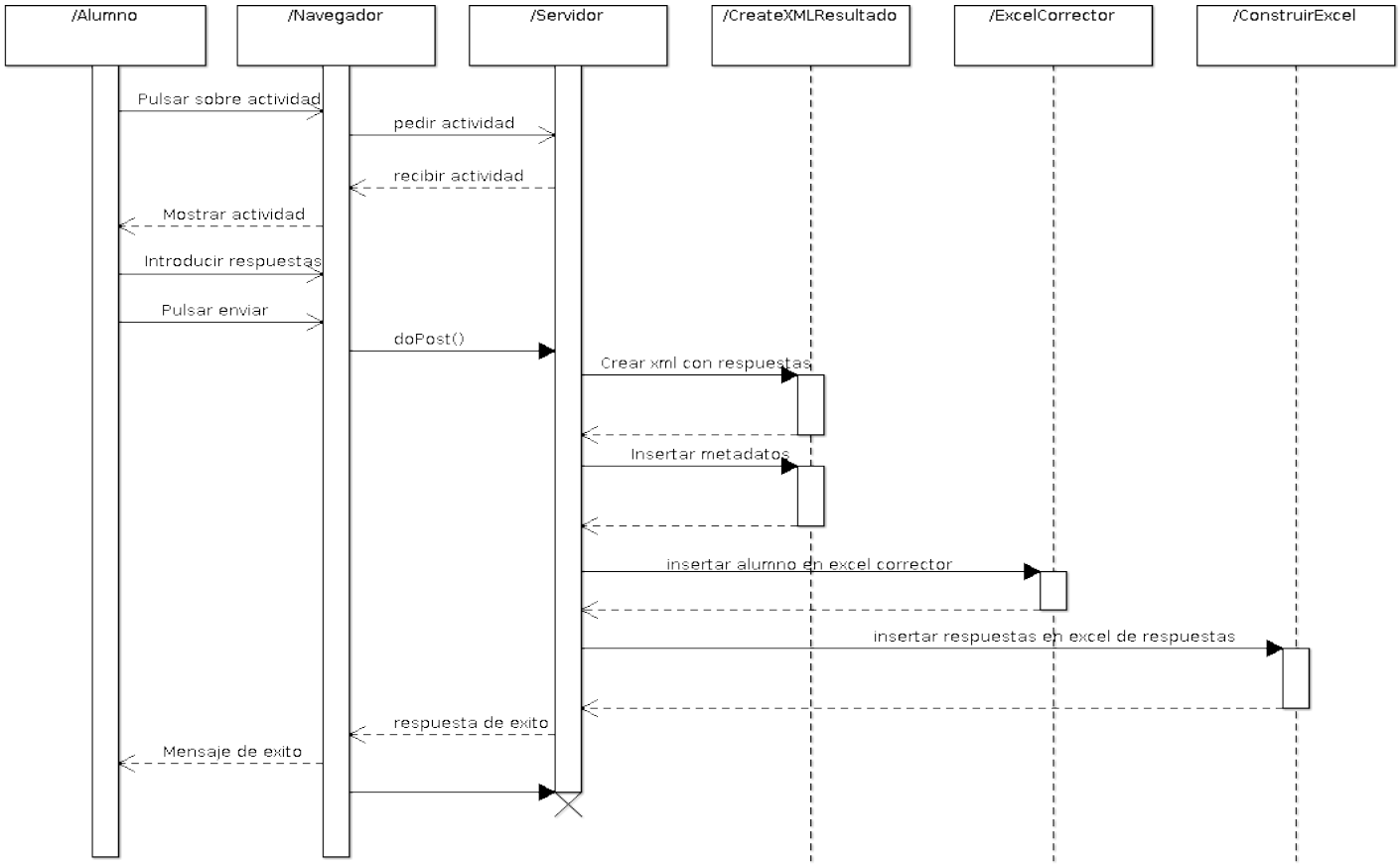


Ilustración 28: Diagrama de secuencia (Realizar actividad)

Una vez que la actividad se ha insertado, los alumnos pueden proceder a realizarla. Se realizará a través del navegador que es usado como interfaz ya que la actividad consiste en un fichero html resultando ser un formulario, este se comunicará con el servidor donde esté alojado la aplicación VideoDataWeb enviándole las respuestas que ha desarrollado el alumno y tras pulsar el botón *Enviar.*

Al enviar las respuestas al servidor, utilizará el Servlet que ha sido creado e insertado a partir de la actividad en el servidor y este se encargará de evaluar las respuestas que se puedan corregir de forma automática (*preguntas de tipo test*) y creará o modificará los siguientes ficheros por cada alumno:

- Crea un fichero xml que contiene los resultados y todas las respuestas del alumno.
- Inserta una nueva fila con todas las respuestas en un fichero Excel que contiene las respuestas dadas por todos los alumnos.

- Inserta una nueva columna con el email del alumno en el Excel corrector que se utilizará para puntuar las preguntas que no son corregidas automáticamente.

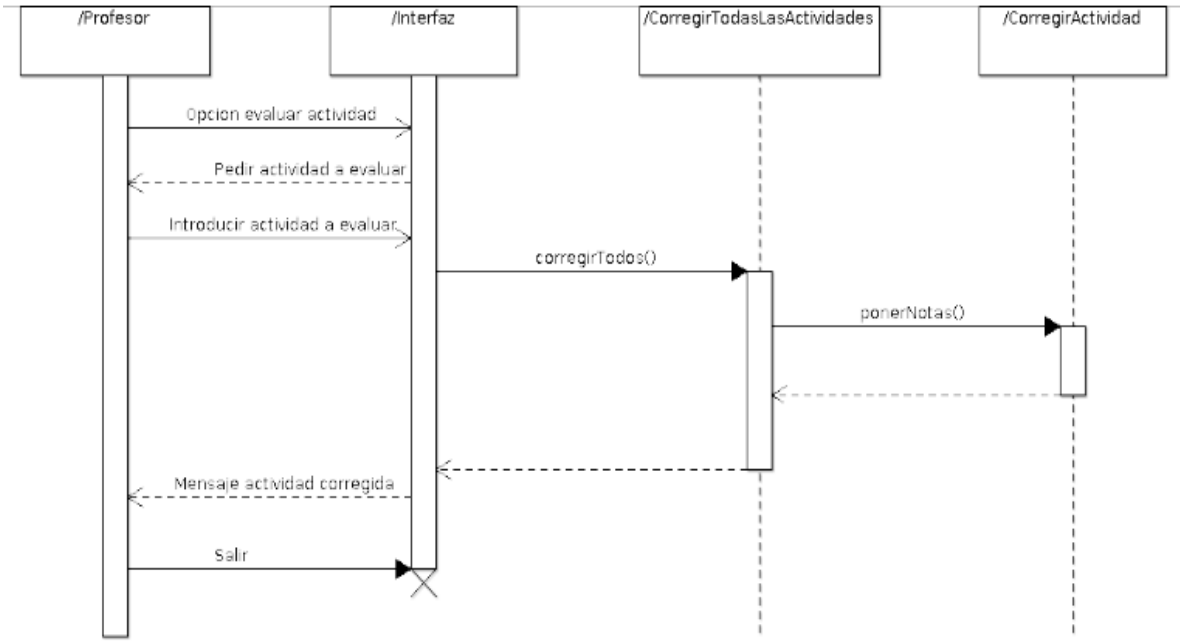


Ilustración 29: Diagrama de secuencia (Evaluar actividad)

Una vez realizada la actividad por parte de los alumnos, el profesor ya puede proceder a la evaluación de los alumnos. El primer paso es rellenar el Excel corrector que contiene las preguntas a evaluar de todos los usuarios que han realizado la actividad.

Después el siguiente paso es seguir la secuencia, seleccionando la opción de evaluación, introduciendo el nombre de la actividad a evaluar y del proceso de evaluación se encarga la clase **CorregirTodasLasActividades** que va corrigiendo a cada alumno usando el Excel corrector.

Tras esto queda modificado el fichero xml de cada alumno que guarda las respuestas y los resultados obtenidos en la actividad. Estos ficheros xml serán los que se guarden en el vídeo que es usado como base de datos.

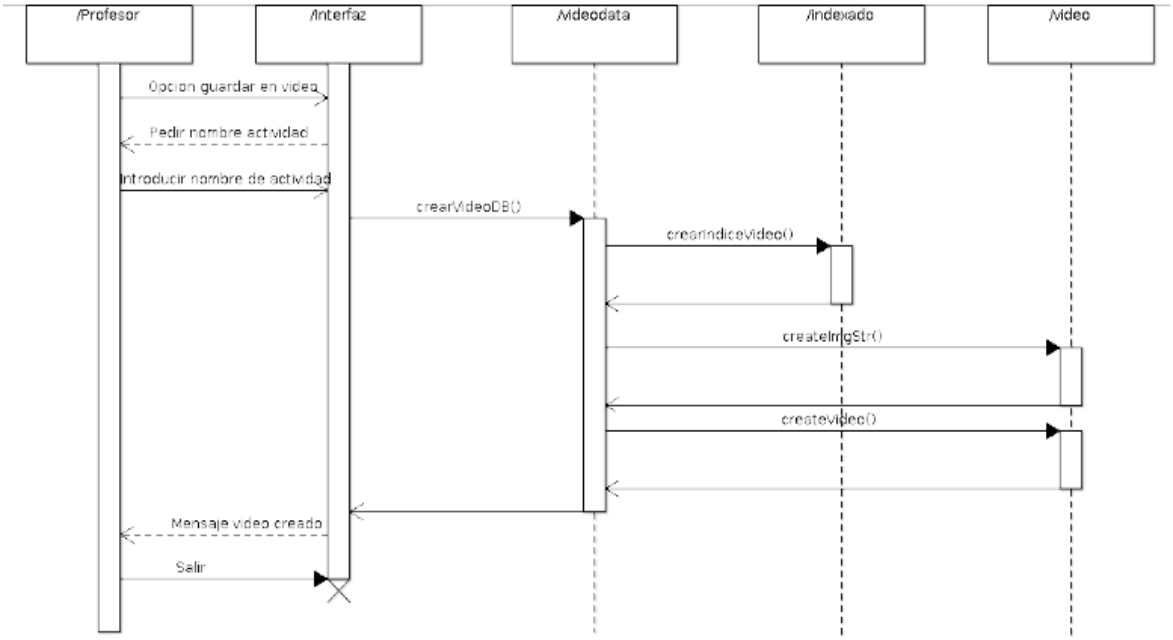


Ilustración 30: Diagrama de secuencia (Guardar en vídeo)

Esta opción crea el vídeo que contiene todos los resultados y respuestas de los alumnos. Para crear el VideoData se selecciona esta opción, se introduce el nombre de la actividad a guardar y se procede a crear el vídeo.

El proceso es bastante complejo y se comentará de forma más detallada en la fase de desarrollo, pero básicamente crea el índice que se utilizará para recuperar la información a partir del vídeo y después procede a crear las imágenes que contienen el contenido de los ficheros xml de resultados y respuestas de los alumnos.

Una vez que está creadas todas las imágenes, se procede a crear el vídeo metiendo todas las imágenes en el vídeo.

Esta es la interacción entre los objetos del módulo VideoData para conseguir usar vídeos como bases de datos, pero esta interacción estará aumentada con mucho más nivel de detalle en el desarrollo debido a que ha resultado ser un proceso complejo.

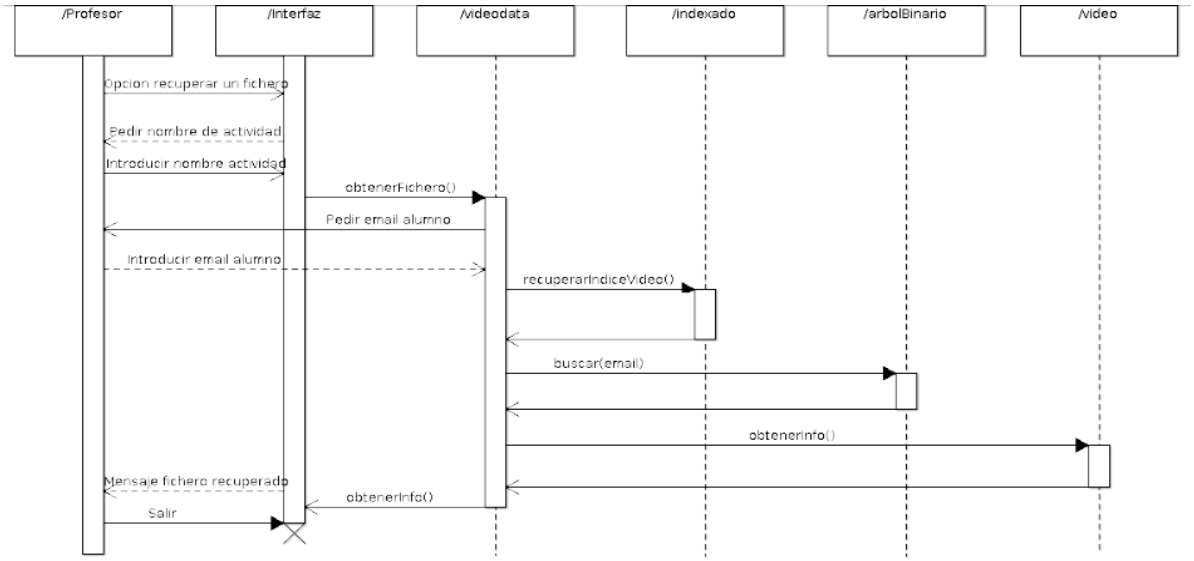


Ilustración 31: Diagrama de secuencia (Recupera un fichero)

Una vez que el vídeo ha sido creado ya tiene guardado toda la información de las respuestas y resultados de los alumnos sobre la actividad. Una opción que se puede utilizar es recuperar un fichero, es decir el fichero xml con respuestas y resultados de un alumno sobre una actividad.

Para realizar este proceso hay que seleccionar esta opción e introducir el nombre de la actividad de la que se quiere recuperar un fichero, después pedirá el identificador del email del alumnos que se desea recuperar.

Una vez introducido el email, se recupera el índice que fue creado junto a la creación del vídeo y se procede a buscar el identificador del email introducido en el índice. Si se encuentra, se obtiene el frame donde se encuentra el fichero del alumno, el píxel del frame donde empieza el fichero y el tamaño del fichero.

Con esos datos se extrae el frame indicado del vídeo y se obtiene la información de este, reconstruyendo el fichero y guardándolo en un fichero xml.

Otra opción es recuperar la actividad, esto significa la recuperación del fichero que contiene las preguntas, posibles respuestas y sus valores. Esta opción es igual que la anterior pero sin pedir el identificador del email ya que tiene un nombre prefijado que coincide con el nombre de la actividad.

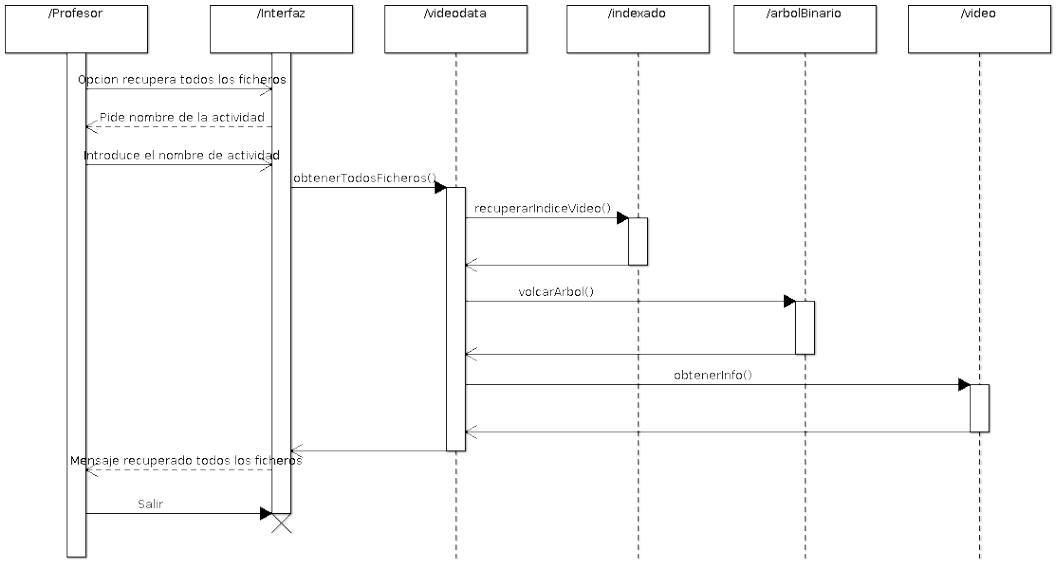


Ilustración 32: Diagrama de secuencia (Recupera todos los ficheros)

Otra opción es recuperar todos los ficheros realizando una especie de carga masiva, el proceso es similar al anterior pero con la diferencia de que no se pide el identificador de ningún alumno ya que se extraerán todos.

Una vez recuperado el índice, se vuelca el árbol y se recupera cada elemento del árbol.

La recuperación de ficheros se puede comprobar una vez realizada en el directorio *directorioProyectoVideoData/VideoDataDB/Recuperado.*

Capítulo 4.- Desarrollo del sistema

Una vez realizado el análisis y desarrollo del sistema, además de tener claro las tecnologías usadas, el flujo y comportamiento del sistema, ya se puede proceder a realizar la fase de desarrollo del sistema.

Estructura del proyecto

El proyecto está formado a partir de un conjunto de proyectos que se interconectan entre sí.

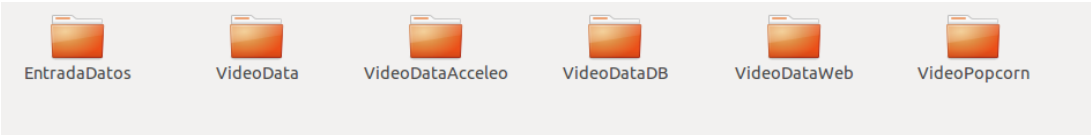


Ilustración 33: Estructura proyecto

El proyecto lo forman los siguientes directorio:

- **EntradaDatos:** es un directorio normal que contiene los ficheros Excel de entrada de las actividades (*/Actividades*) y los hipervídeos (*/Popcornjs*).
- **VideoData:** es un proyecto de tipo *EMF* que se utiliza como proyecto principal ya que realiza la creación de contenido, la inserción de actividades y la evaluación de estas. Es un proyecto EMF porque contiene el metamodelo y también contendrá los modelos que en este caso serán las actividades.
- **VideoDataAcceleo:** es un proyecto de tipo *Acceleo* que realiza la transformación de modelo a texto para generar el código de las actividades.
- **VideoDataDB:** es el directorio donde se almacenan todos los datos generados a partir de la realización de las actividades por parte de los alumnos.
- **VideoDataWeb:** es un proyecto de tipo *Dynamic Web Project* que contiene la aplicación web sobre la que se corren los hipervídeos y las actividades.
- **VideoPopcorn:** es un directorio usado como copia de seguridad de los hipervídeos.

Estructura VideoData

Hay que destacar la importancia del proyecto VideoData ya que es el proyecto principal y tiene una estructura compleja.

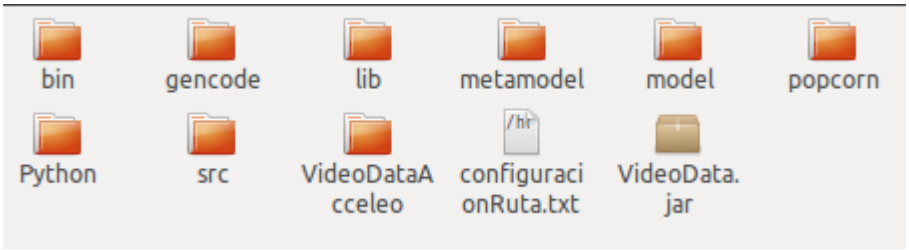


Ilustración 34: Estructura proyecto VideoData

Contiene los siguientes ficheros y directorios para su correcto funcionamiento:

- bin: contiene los ficheros binarios.
- gencode: directorio donde se almacena de forma temporal el código generado para las actividades usando Acceleo.
- lib: contiene las librerías básicas que se usan en el proyecto. Para manejar los ficheros Excel y XML.
- **metamodel**: contiene el metamodelo de las actividades.
- **model**: directorio que almacena las actividades después de su creación. Están en formato xmi y son modelos del metamodelo desarrollado.
- **popcorn**: contiene el esquema para validar los xml de hipervídeos creados y el fichero xslt para realizar la transformación del xml al html que representa el hipervídeo.
- **Python:** es el directorio del módulo de VideoData (almacenamiento en vídeo) y contiene todo el código fuente Python que trabaja con vídeos para el almacenamiento y extracción de información. Se describirá más adelante de forma detallada ya que es muy importante.
- **src:** contiene los ficheros fuentes de la creación de contenido y evaluación de

una forma organizada en paquetes tal como se puede apreciar en su Modelo Conceptual.

- **VideoDataAcceleo:** es muy importante para el funcionamiento del fichero VideoData.jar, es el contenido del directorio *VideoDataAcceleo/bin* y debe ser copiado cada vez que se modifique el proyecto Acceleo.
- **configuracionRuta.txt:** fichero de configuración que contiene la ruta a dicho directorio. Está detalladamente en el apartado de Configuración del proyecto en el manual del usuario.

Compilado de la aplicación

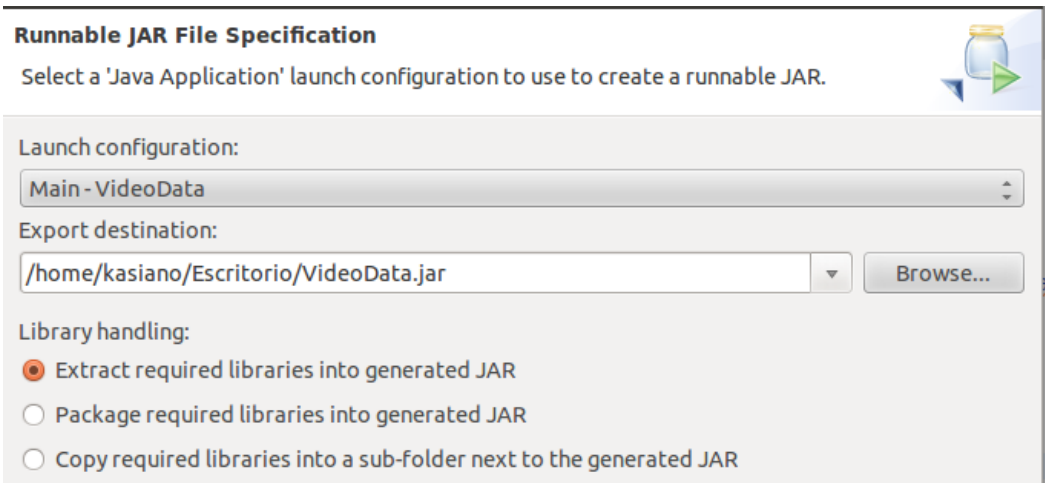


Ilustración 35: Compilando aplicación

- **VideoData.jar:** es el ejecutable del módulo de creación de contenido y evaluación. Si se modifica alguna parte del código fuente hay que volverlo a crear. Para ello hay que pulsar sobre el proyecto VideoData en eclipse con *botón derecho → Export → Runnable JAR File →* y seleccionar las opciones marcadas en la imagen anterior y la ruta de destino.

Python VideoData

El directorio *VideoData/Python* es muy importante debido a que contiene todo el código fuente relacionado con el tratamiento de vídeos como base de datos y la recuperación de la información.

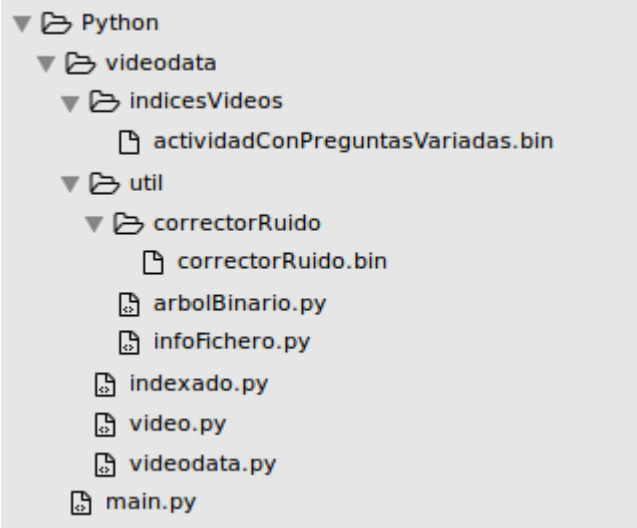


Ilustración 36: Estructura Python VideoData

La estructura es la siguiente:

- **main.py**: contiene la interfaz desde la que el profesor crea los vídeos como almacenamiento.
- **videodata**: directorio que contiene los siguientes elementos:
    - **videodata.py**: contiene toda la lógica principal, el fichero main.py hace llamadas a métodos que se encuentran en este módulo.
    - **video.py**: contiene el código fuente relacionado con la creación y obtención de vídeos. Es el módulo que utiliza OpenCV.
    - **indexado.py**: realiza los índices de los vídeo y los recupera.
    - **indicesVideos**: directorio donde se guardan los índices. Son ficheros binarios con el mismo nombre que la actividad guardada en vídeo que contienen un árbol binario donde cada elemento es un fichero dentro del vídeo.
    - **util:** directorio que contiene los siguientes elementos:

- **arbolBinario.py:** es la implementación del árbol binario, es la estructura de datos utilizada como índice.
- **infoFichero.py**: es el contenido de cada nodo del árbol binario y contiene la información necesaria para localizar un fichero en un vídeo.
- **correctorRuido**: contiene un fichero para corregir el ruido de los frames en la creación del vídeo.

Creación de contenido

La creación de contenido abarca dos elementos muy importantes que pueden complementarse entre ellos como son las actividades y los hipervídeos.

Modelado de actividades

Las actividades consisten en un conjunto de preguntas que pueden ser de distintos tipos. En la implementación de actividades se ha utilizado el modelado de software debido a que permite la descomposición y modularización de las actividades, además de mejorar la evolución, mantenimiento y reusabilidad de las actividades. Otro factor clave es que permite la generación de código automático a partir de los modelos.

En el modelado de las actividades se ha usado EMF y ecore para la implementación, por ello el primer paso ha sido crear el proyecto de tipo EMF llamado VideoData. Este proyecto es el principal porque también se ocupa de la entrada de datos, la creación de hipervídeos y la evaluación no automática.

Metamodelo

Como todo modelado de software, tras realizar un análisis de las características que deben tener las actividades hay que proceder al desarrollo del metamodelo.

Hay que crear un '*ecore model'*, en este caso se llama **ActividadMM**, se encuentra en ***proyectoVideoData/VideoData/metamodel*** y son dos ficheros con distinta extensión:

- *ecore:* es el metamodelo en sí. Contiene toda la estructura y toda la lógica.
- *ecorediag:* es la representación gráfica del metamodelo, es decir el editor del

metamodelo. Cuando estemos trabajando con este editor es conveniente tener cerrado el fichero .ecore para que se actualiza de forma correcta.

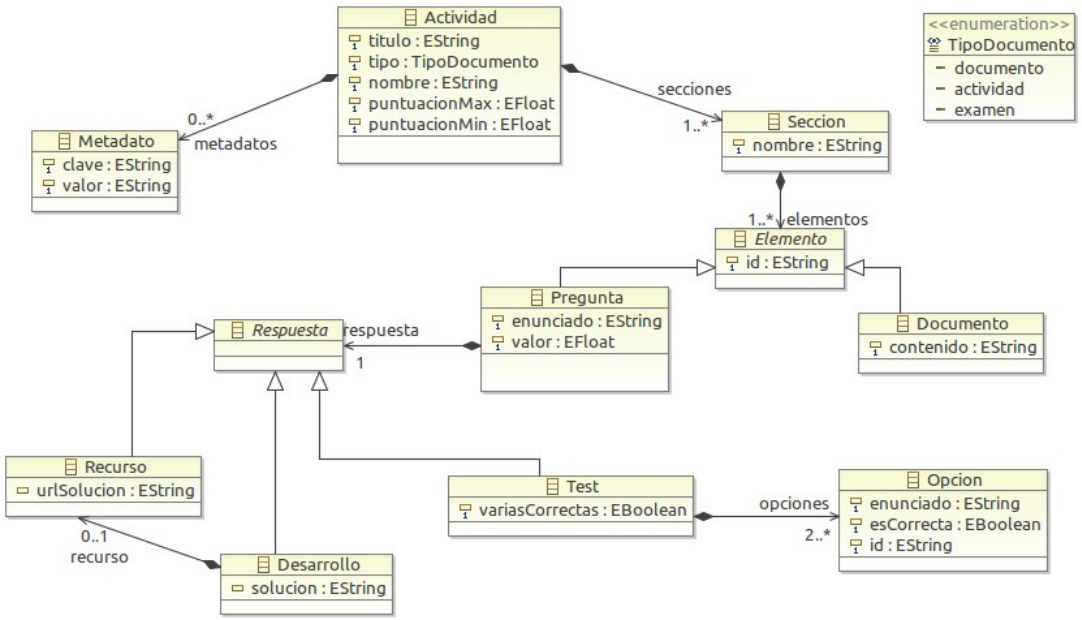


Ilustración 37: Metamodelo de actividad

En la imagen anterior se puede ver el fichero ecorediag que contiene el metamodelo mediante la representación gráfica que se utiliza como editor. Es un metamodelo simple donde **el elemento root es *Actividad*** y puede o no contener *metadatos* <clave,valor>. Además de los metadatos, **es obligatorio que la actividad tenga al menos una S*ección***.

Las **secciones contendrán un conjunto de elementos** que pueden ser de tipo Documento que únicamente es un texto o Pregunta que es un elemento más complejo. Y una ***Pregunta*** contiene el enunciado y el valor, además de una ***Respuesta*** que puede ser de varios tipos:

- ***Recurso:*** contiene un campo de texto para introducir una url al recurso pedido.
- ***Desarrollo:*** contiene un campo de texto para introducir la teoría que sea pedida en el enunciado de la pregunta. Puede tener también una respuesta de tipo recurso opcionalmente.
- ***Test:*** se utiliza para preguntas de tipo test, el valor *variasCorrectas* indicará en el modelo si esa respuesta tiene una solución posible (test simple) o puede tener varias (test múltiple). El test debe tener como mínimo dos ***opciones*** que

contienen un id, el enunciado y si es o no correcta dicha opción.

Restricciones

Una vez desarrollado el metamodelo, es necesario el uso de restricciones para controlar y evitar futuros errores como pueden ser elementos con el mismo identificador o que la puntuación máxima de la actividad no coincida con la suma de puntuación de cada pregunta.

Las restricciones al tratarse de un metamodelo de tipo ecore se realizarán usando **OCL**, para poder empezar a definir restricciones hay que pulsar sobre el metamodelo con *botón derecho → Open with → OCLinEcore* y aparecerá el código fuente del metamodelo en java donde se pueden definir las restricciones.

Hay que tener en cuenta que cada restricción es definida en base a un contexto concreto, es decir que afecta a un elemento del metamodelo específico.

Las **restricciones en el contexto Actividad son**:

- La puntuación mínima de la actividad debe ser menor a la máxima.

  ```
  invariant puntuacionMinExcesiva: self.puntuacionMax >= self.puntuacionMin;
  ```

- La puntuación máxima debe coincidir con la suma del valor de cada pregunta.

  ```
  invariant puntuacionMaxIncorrecta:
  self.puntuacionMax = Pregunta.allInstances()->iterate(p : Pregunta; sum : Real = 0 | sum + p.valor);
  ```

- Tanto la puntuación máxima como la mínima de la actividad debe ser positiva.

  ```
  invariant puntuacionesMaxNegativa: self.puntuacionMax >= 0;
  invariant puntuacionesMinNegativa: self.puntuacionMin >= 0;
  ```

- No pueden existir dos metadatos con la misma clave.

  ```
  invariant metadatosIguales: self.metadatos->forAll(m1 : Metadato, m2 : Metadato | m1 <> m2 implies m1.clave <> m2.clave);
  ```

- No puede haber metadatos que tengan como clave el nombre título o nombre.

```
invariant nombreMetadatoNoPermitido: self.metadatos->forAll(m1 : Metadato | m1.clave <> 'titulo' and m1.clave <> 'nombre');
```

- Si la actividad tiene varias secciones estas no pueden tener el mismo nombre.

```
invariant seccionesIguales: self.secciones->forAll(s1 : Seccion, s2 : Seccion | s1 <> s2 implies s1.nombre <> s2.nombre);
```

- Si el tipo de la actividad es documento, este no puede tener ninguna pregunta.

```
invariant actividadTipoDocumentoConPreguntas: let numPreguntas : Integer = Pregunta.allInstances()->size() in if self.tipo = TipoDocumento::documento then numPreguntas = 0 else numPreguntas > 0 endif;
```

La restricción en el contexto Seccion es:

- La sección no puede tener elementos (pregunta o documento) con un mismo id.

```
invariant preguntasConMismoId: self.elementos->forAll(e1 : Elemento, e2 : Elemento | e1 <> e2 implies e1.id <> e2.id);
```

La restricción en el contexto Pregunta es:

- No pueden existir preguntas cuyo valor sea negativo.

```
invariant preguntaConValorNegativo: self.valor >= 0;
```

Las restricciones en el contexto Test son:

- Un test que sea de tipo simple solo puede tener una opción correcta, si el test es de tipo múltiple este debe tener una o varias opciones que sean correctas.

```
invariant numIncorrectoDeRespuestasCorrectas: let numRespuestasCorrectas : Integer = self.opciones->iterate(o : Opcion; numCorrectas : Integer = 0 | if o.esCorrecta then numCorrectas + 1 else numCorrectas endif) in if not self.variasCorrectas then numRespuestasCorrectas = 1 else
```

```
numRespuestasCorrectas >= 1 endif;
```

- Las opciones de un test deben tener distinto id.

```
invariant opcionesTestConMismoId: self.opciones->forAll(o1 :
Opcion, o2 : Opcion | o1 <> o2 implies o1.id <> o2.id);
```

- Las opciones de un test deben tener distinto enunciado.

```
invariant opcionesTestConMismoEnunciado: self.opciones-
>forAll(o1 : Opcion, o2 : Opcion | o1 <> o2 implies
o1.enunciado <> o2.enunciado);
```

Modelo

Los modelos serán las actividades que se crean a partir de la hoja de cálculo. Puede decirse que es la sintaxis concreta de una actividad cuando el metamodelo es una sintaxis abstracta. Los modelos son guardados en la ruta ***proyectoVideoData/VideoData/model***.

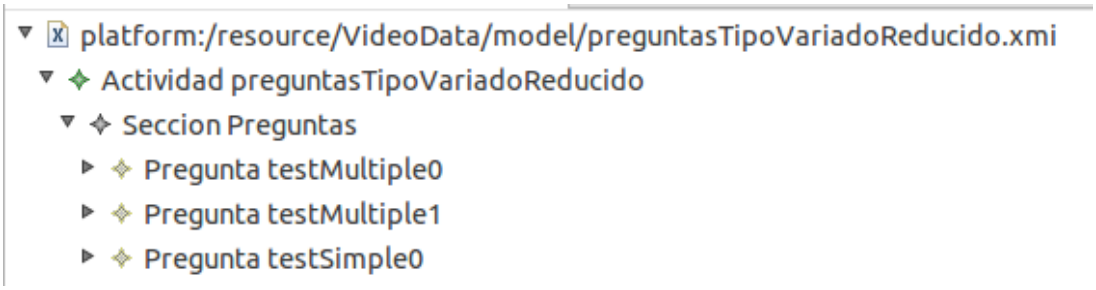


Ilustración 38: Vista de un modelo

En la imagen se puede apreciar como se visualiza un modelo, mediante una estructura de árbol aunque **el formato del modelo es xmi** que es un formato xml. Por lo tanto, como el objetivo es poder crear actividades (modelo) a partir de hojas de cálculo, se crea de forma automática el fichero en formato xmi. De esto se encarga la clase **Excel2Model** que recibe el fichero excel y crea un modelo. Esta clase utiliza la clase **GestionTag** para filtrar el fichero de entrada e incluir solo las preguntas que pertenezcan a los tags seleccionados y creando en el modelo una sección por cada tag diferente.

```
private static void crearModeoDesdeExcel() throws IOException  {
	// obtenemos el fichero
	File inputWorkbook = new File(inputFile);
	Workbook w;
	List<String> row;
```

```
// obtenemos el excel y la hoja
	w = Workbook.getWorkbook(inputWorkbook);
	Sheet sheet = w.getSheet(0);
// creamos los metadatos a partir del otro excel de metadatos
	createMetadatas();

// cada linea del excel se extrae y se procesa para crear el modelo
	for (int numRow = 1; numRow < sheet.getRows(); numRow++) {
		row = getRow(sheet, numRow);
		if (preguntaValida(row))
			crearModelo(row);
	}
}
```

El algoritmo abre la hoja de cálculo y lo lee línea a línea obteniendo su contenido, se comprueba que es una línea válida comprobando que el formato es correcto y pertenece a uno de los tags seleccionados. Si es correcta se crea esa pregunta en el modelo y una vez se ha terminado de ejecutar este algoritmo se guarda el fichero en formato xmi.

Generación de código (Acceleo)

Después de tener ya los modelos y un sistema sencillo para crearlos, el último paso consiste en utilizar esas actividades o modelos y transformarlas en código. Para esta función que permite el modelado llamado modelo a texto *m2t* se usará **Acceleo**.

Es necesario utilizar un proyecto de tipo '*Accelo project*' desde donde se ejecutará la plantilla de transformación, en este caso el nombre del proyecto es **VideoDataAcceleo**. La ruta es ***proyectoVideoData/VideoDataAcceleo***.

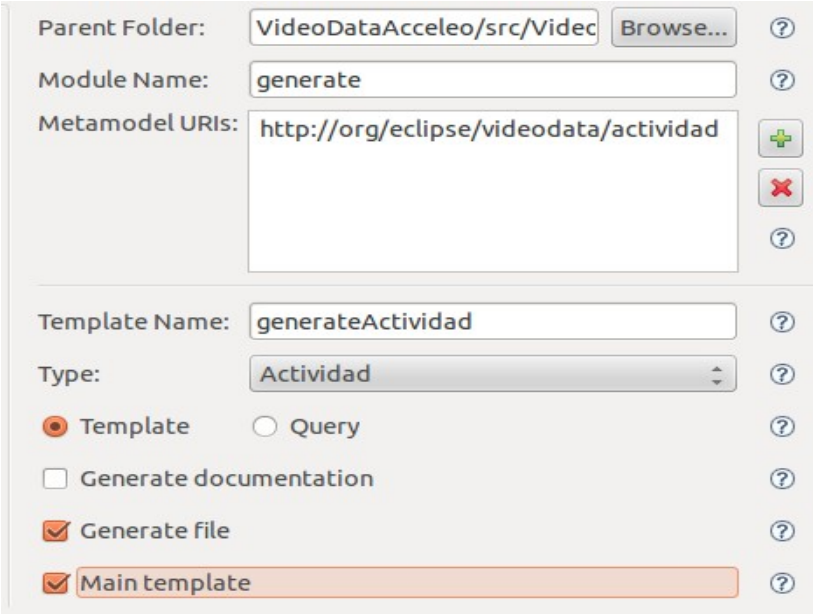


Ilustración 39: Template Acceleo

Es importante seleccionar la URI del metamodelo que se haya indicado en el fichero ecore y marcar las opciones *Generate file* y *Main template*. En este fichero creado se desarrollará el código a generar de forma automática dado cualquier modelo. Es un sistema de templates que tiene una sintaxis propia y que se basa en recorrer la estructura del metamodelo por cada elemento.

```
[template public generateActividad(act : Actividad)]
[comment @main/]

[generateHTML(act) /]
[createEnunciadosEnExcel(act) /]
[generateServlet(act) /]

[/template]
```

El template generará tres ficheros diferentes que se usarán para representar la actividad:

- Fichero html: contiene la visualización de la actividad desde un navegador, consiste en un formulario.
- Fichero enunciados excel: es un fichero java que contiene una clase para crear un fichero excel donde la primera línea contiene en cada celda el enunciado de una pregunta. Este fichero se usará para guardar las respuestas de todos los alumnos para que el profesor tenga una vista generalizada de las respuestas de todos los alumnos.

- Servlet: es un fichero java que contiene la lógica de la actividad debido a que las actividades se incluirán en una aplicación web para que puedan ser realizadas por los alumnos. La lógica declarada es ejecutada en el servidor cuando un alumno envía la actividad realizada y corrige las preguntas de tipo test además de guardar los resultados y respuestas del alumno.

Se realiza una llamada a este template Acceleo usando la clase **Model2Code** que simplemente aplica el template generando los fichero indicados sobre el modelo o actividad introducida por parámetros.

Desarrollo de hipervídeos

En esta sección se comentará como se ha desarrollado todo lo relacionado con el hipervídeo. En su desarrollo se utilizarán tecnologías XML, concretamente los hipervídeos serán en un inicio un fichero xml que tendrá que ser validado por un fichero xsd y se es un fichero válido se procederá a la transformación del xml en un html que contendrá la visualización del hipervídeo con todos los eventos que tenga.

Creación del esquema

El primer paso es definir el esquema con el que debe validarse el fichero xml. Por este motivo se ha desarrollado un esquema en formato **XSD**. El esquema se encuentra en la ruta ***proyectoVideoData/VideoData/popcorn***.

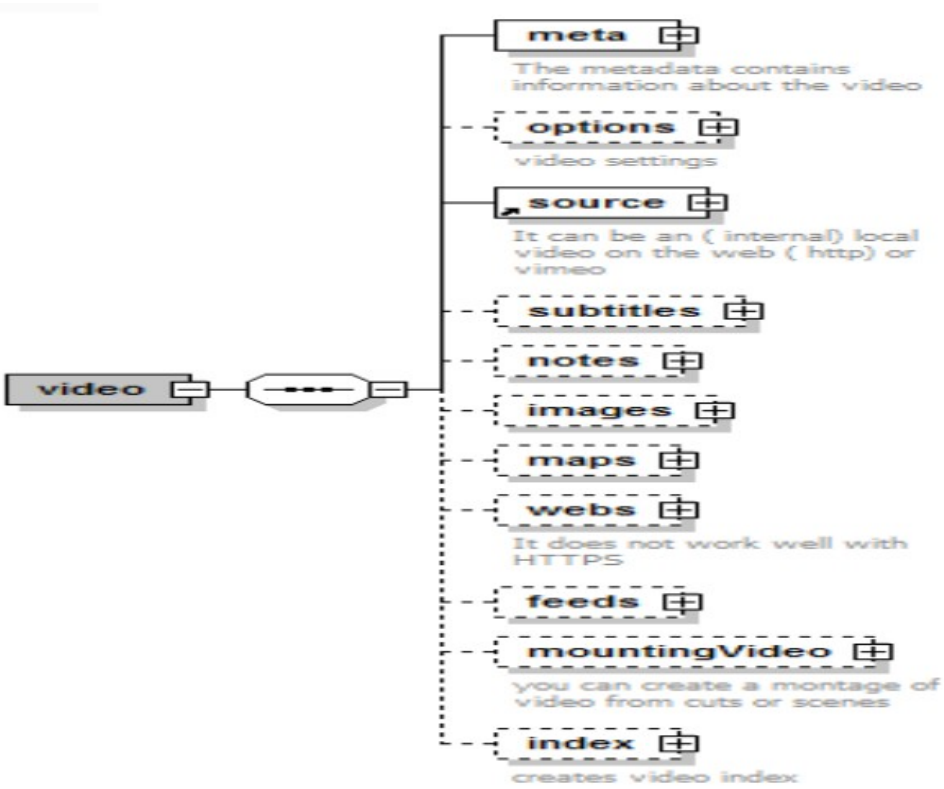


Ilustración 40: Esquema del hipervídeo

En la ilustración anterior se puede ver el esquema que debe validar la representación de los hipervídeos. Contiene los elementos que puede tener el hipervídeo donde los únicos campos obligatorios son *meta* y *source*. Es esquema es muy importante ya

que contiene la estructura que tendrán los ficheros xml que representen los hipervídeos, se ampliará el esquema en las siguientes imágenes.

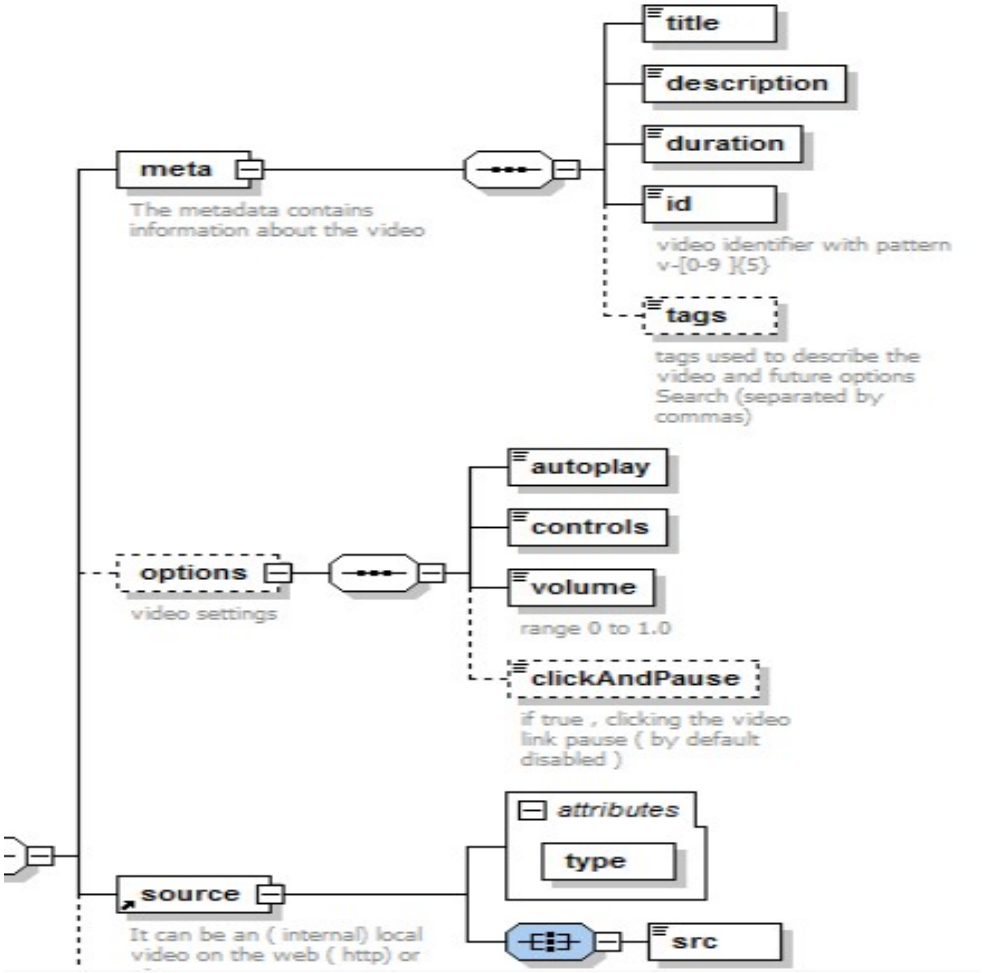


Ilustración 41: Esquema hipervídeo detallado (parte I)

En esta imagen se pueden ver los elementos de configuración del hipervídeo, estos campos son rellenados desde la hoja de cálculo de metadatos y contiene el elemento *meta* con elementos como el título y el nombre. Además el elemento *options* es muy importante ya que permite configurar aspectos claves en la reproducción del hipervídeo como puede ser si se reproducirá de forma automática, si se mostrará el cuadro de control sobre el vídeo, si al pulsar sobre un enlace se pausará el vídeo o el volumen.

Por último el elemento clave es *source* ya que indica el tipo de vídeo (*internal* o *http*) y el nombre del vídeo o url.

Los siguientes elementos ya forman parte de la estructura del hipervídeo debido a

que son los eventos que se ejecutarán en función del tiempo de ejecución del vídeo.

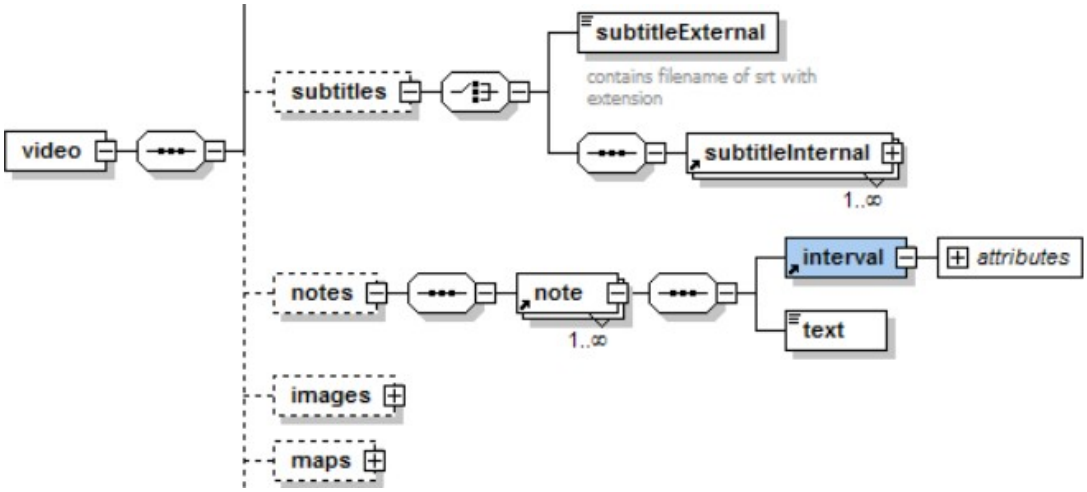


Ilustración 42: Esquema hipervídeo detallado (parte II)

En la imagen se puede apreciar los algunos eventos usados como los subtítulos junto a sus dos modalidades *(externos* o *internos).* También las notas, imágenes y mapas que tienen la misma estructura siendo una secuencia de elementos y cada uno tiene un intervalo *(inicio* y *fin)* y un texto.

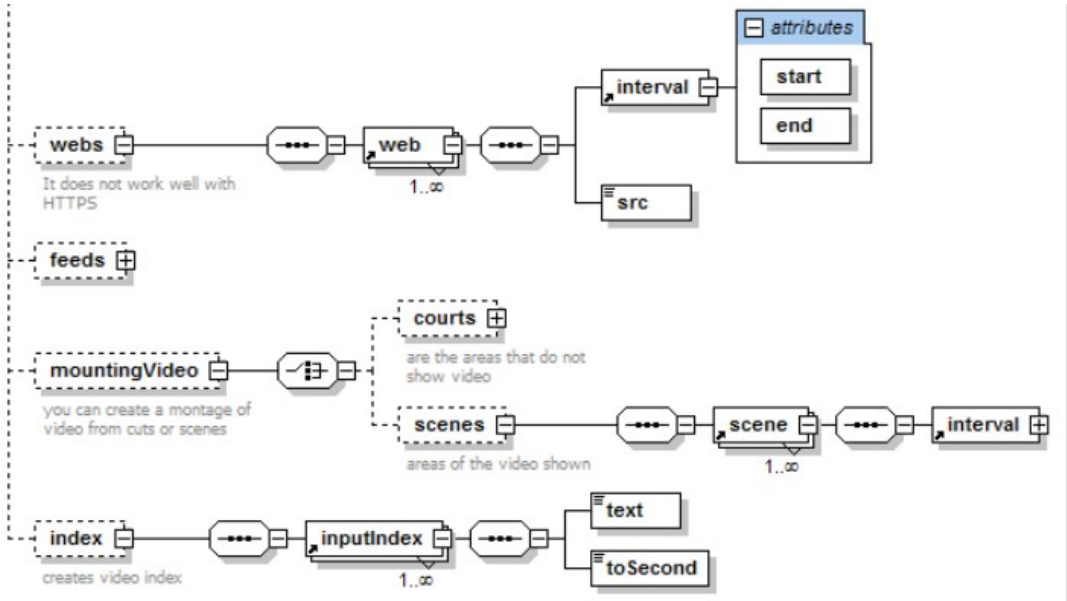


Ilustración 43: Esquema hipervídeo detallado (parte III)

La última parte del esquema contiene un elemento muy importante como es **web** ya que **en este campo se incluirán las actividades** generadas siendo src la url al

recurso. Además están los feeds con una estructura idéntica al de web.

También un elemento muy importante es **scenes** ya que define las escenas a mostrar del vídeo. En esta esquema también está definido otra forma a la creación del montaje usando cortes o escenas eliminadas pero no se utilizarán en la creación de hipervídeos desde hojas de cálculo.

Por último el índice que establece el texto a mostrar y el segundo donde se redirige.

Representación de los hipervídeos

La representación de los hipervídeos son ficheros xml que corresponden al esquema indicado con anterioridad. Estos ficheros xml son creados de forma automática a partir de unas hojas de cálculo como entrada de datos. Estos ficheros se guardan en la ruta ***proyectoVideoData/VideoData/popcorn/recursos***.

La clase encargada de la creación del xml a partir del fichero de entrada es **Excel2XMLPopcorn**.

```
private static void crearXMLDesdeExcel() throws IOException  {
	File inputWorkbook = new File(inputFile);
	Workbook w;
	List<String> row;
	w = Workbook.getWorkbook(inputWorkbook);
	Sheet sheet = w.getSheet(0);
// creamos los metadatos que contienen informacion muy importante y es
obligatorio
	crearMetadatos();
	// creamos el tiempo con la primera linea del excel
	row = getRow(sheet, 1);
	crearTiempo(row);
	// creamos las escenas con la octava linea del excel
	row = getRow(sheet, 8);
	crearEscenas(row);

	// para cada linea creamos los eventos de un determinado tipo
	for (int numRow = 2; numRow < sheet.getRows(); numRow++) {
		row = getRow(sheet, numRow);
		if (!row.isEmpty())
			crearXML(row, numRow);
	}
}
```

Primero se crean los metadatos a partir de una hoja de cálculo, después a partir del excel que contiene la estructura del hipervídeo se crean primero los tiempos donde se activarán los eventos y las escenas del vídeo, después se recorre cada línea creando un evento determinado en función del fichero de entrada.

Generación del hipervídeo

Una vez que se tienen los fichero xml con la representación del hipervídeo puede procederse a generar el hipervídeo para que pueda ser visualizado. Al emplear tecnologías web hay que generar un fichero html que se podrá reproducir en cualquier navegador. Este fichero se encuentra en la ruta ***proyectoVideoData/VideoData/popcorn***.

La tecnología empleada será **XSLT** debido a que resulta una herramienta perfecta en transformaciones xml. Para ello se ha desarrollado un fichero xsl que contiene las transformaciones a realizar sobre el xml para poder convertirlo en un fichero html.

```
<xsl:template match="//note">
	pop.footnote({
		start: <xsl:value-of select="./interval/@start"/>,
		end: <xsl:value-of select="./interval/@end"/>,
		text: "<xsl:value-of select="./text/text()"/>",
		target: 'zoneNote'
	});
</xsl:template>
```

Ese es un trozo del fichero xsl, para su desarrollo hay que tener nociones en esta tecnología además de saber programar en html y JavaScript debido a que **en la realización de los hipervídeos se utiliza popcornjs que es una librería JavaScript que permite incrustar eventos al vídeo para crear un contenido enriquecido**.

La clase **XML2HTMLViaXLST** es la encargada primero de **validar el xml usando la clase ValidadorXML** que valida el fichero contra el esquema y **después utiliza el fichero xsl para transformar el fichero y conseguir el html que permite visualizar el hipervídeo**.

Evaluación de actividades

El proceso de evaluación consiste en visualizar el hipervídeo y realizar las actividades indicadas, esto se realiza a través de una aplicación web J2EE llamada VideoDataWeb. Para cada una de estas actividades se producen tres fases de evaluación:

- **Evaluación automática:** esta evaluación es realizada en la lógica del servidor mediante el servlet que se ha generado de la actividad. Consiste en evaluar las preguntas de tipo test de forma automática.
- **Guardado de datos:** esta parte no es una evaluación como tal, pero también es realizada por el servlet y consiste en guardar los datos del alumno. Estos datos consisten en los resultados de la evaluación automática y todas las respuestas. Se realizan tres acciones:
    - *Guardar fichero del alumno:* se guarda un fichero en formato xml con las respuestas y resultados por cada alumno.
    - *Incluir respuestas en excel:* por cada actividad existe un fichero excel que contiene en cada fila las respuestas realizadas por un alumno. De esta forma el profesor puede visualizar las respuestas de todos los alumnos de forma unificada.
    - *Incluir alumno en excel corrector*: existe un **excel corrector que contiene en cada fila el id de las preguntas a corregir de forma manual y en la primera fila en cada celda el email del alumno**.
- **Evaluación manual:** esta evaluación es realizada por el profesor sobre las preguntas de la actividad que no pueda ser corregida de forma automática y **no es realizada ya en la aplicación web**, es decir evaluará las preguntas de tipo desarrollo y recurso. Usa el **excel corrector** rellenando e indicando la puntuación obtenida por cada alumno en cada pregunta.

  Una vez rellenado por el profesor el excel corrector se utiliza la clase **CorregirTodasLasActividades** que va leyendo el corrector y modificando el fichero xml de cada alumno que contiene los resultados y las respuestas.

Almacenamiento en vídeo (VideoData)

En esta sección se explicarán las claves del desarrollo relacionado con el almacenamiento de información en vídeo. Ha resultado un desarrollo complejo que ha dado muchos problemas aunque estos serán comentados en la sección **Problemas durante el desarrollo**.

La tecnología utilizada en este módulo es OpenCV combinado con Python 2.7 y el proceso consiste en guardar los resultados de los alumnos que son ficheros xml en un vídeo para ser usado como base de datos.

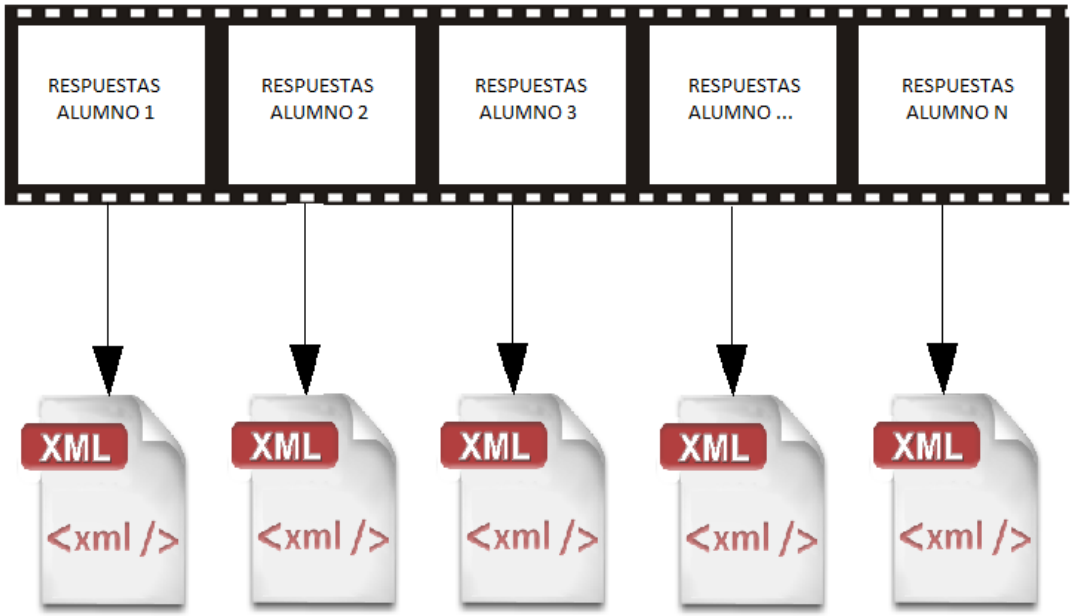


Ilustración 44: VideoData con resultados

De forma que **un frame contenga un trozo de un fichero, un fichero o varios**, estando **indexados por el nombre del fichero para poder conseguir un acceso rápido y eficiente a cada fichero usando el vídeo**.

La eficiencia es posible en el uso de vídeos siempre que la tecnología usada permita **acceder a un frame concreto sin tener la necesidad de recorrer todos los frames anteriores**.

También **es fundamental que los frames del vídeo no tengan pérdida** debido a que estamos tratando con información textual ya que un píxel diferente al deseado significará un fichero diferente al guardado en el vídeo.

Creación del vídeo de almacenamiento

El proceso de crear el vídeo ha resultado ser complejo. Ha habido bastantes problemas aunque se ampliarán en la sección *Problemas durante el desarrollo*, pero básicamente **los principales problemas han sido relacionados con los codecs usados en el vídeo** porque aunque se utiliza un codec *lossless* (*sin pérdidas*) los píxeles que se guardan no son idénticos a los obtenidos después.

Después de analizar los codec lossless compatibles con OpenCV se comprobó que el codec HFYU por cada píxel guardado siempre recupera un valor de píxel diferente pero siempre el mismo.

En la creación del vídeo se guardará en cada píxel de cada frame un carácter de un fichero, debido a esto cada píxel contendrá un Byte. Entonces al guardar el byte 99 'c' siempre se recuperaba el byte 97 'a'. Se observó que el desplazamiento era de +1 byte o +2 byte de tal forma que se decidió emplear un segundo frame por cada frame de información que contendrá el desplazamiento que hay que aplicar para corregir el vídeo.

Es decir, cada frame de información va a tener un frame corrector con la compensación del 'ruido' para reconstruir el frame original guardado y obtener la información sin pérdida. El frame corrector tendrá en cada píxel un valor 100 si la compensación es +1 o un valor 200 si la compensación es +2.

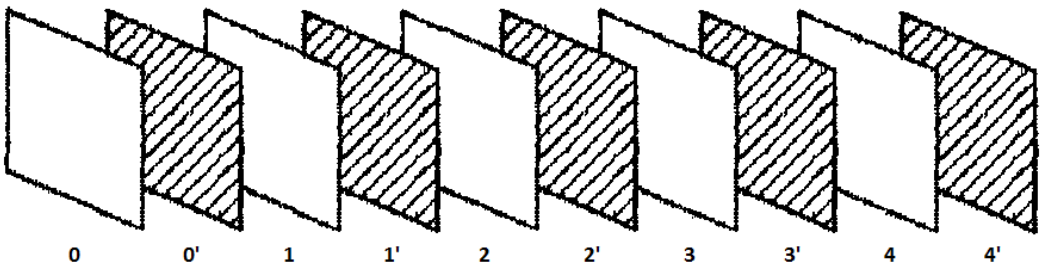


Ilustración 45: Frames del vídeo

De esta forma **los frames 0,1,2,... contendrán los ficheros de resultados** de los alumnos y **los frames 0',1',2',... tendrán la compensación del desvío del codec para reconstruir los ficheros sin pérdida**.

```
def crearVideoDB(nombreModelo):
  rutaVideo = "../../VideoDataDB/VideosDB/{0}.avi".format(nombreModelo)
  listaFicheros = getFicheros("../../VideoDataDB/Respuestas/
                                                {0}/*.xml".format(nombreModelo))
  if len(listaFicheros) > 0:
   CORRECTORRUIDO = video.recuperarCorrectorRuido("videodata/util/correctorRuido")
    while numFich != -1:
      # obtenemos la cadena a partir de los ficheros
      cad, numFich, posIni = construirTrozo(listaFicheros, numFich, posIni)
      lon = len(cad)
      # creamos la imagen usando la cadena construida a partir de los ficheros
      video.createImgStr(nombreModelo, cad, numImgGeneradas, CORRECTORRUIDO)
      numImgGeneradas+=1

     # finalmente se crea el video que usara las imagenes creadas a partir de los
ficheros de resultados
    video.createVideo(nombreModelo, numImgGeneradas, rutaVideo)
```

El algoritmo principal encargado de esta tarea se encuentra en el fichero **videodata.py** y dado el nombre de un modelo o actividad recupera los ficheros de respuestas en una lista, después recupera el corrector de ruido que es usado para crear los vídeos y que se explicará después.

Desde este método se llama a la función *construirTrozo* que construye una cadena del tamaño de un frame (148x100) es decir 14800 caracteres. Esta cadena es construida a partir de las respuestas de los alumnos sobre la actividad.

El siguiente paso es una vez construida esa cadena se crea una imagen donde cada carácter es un píxel de la imagen. Después de crear todas las imágenes necesarias para guardar los ficheros de todos los alumnos, se procede a crear el vídeo que consiste en introducir las imágenes creadas en el vídeo llamando a *createVideo()*.

```
def createVideo(nombreModelo, numFrames, srcVideo='frames/videoBD.avi'):
  # lista de imagenes donde se guardan
  images = []
  # por cada imagen
  for x in range(numFrames):
    srcImg = "../../VideoDataDB/Respuestas/{0}/{1}.png".format(nombreModelo, x)
                        srcImgRuido      =      "../../VideoDataDB/Respuestas/{0}/
{1}ruido.png".format(nombreModelo, x)

    images.append(cv2.imread(srcImg))
    images.append(cv2.imread(srcImgRuido))
  # se calculan las dimensiones
  height , width = images[0].shape[:2]

  # se inicializa el codec
  fourcc = cv2.cv.CV_FOURCC(*'HFYU') # -1 o -2 valores menos
 # se indican los fps del video, recomendado hasta 100
  fps = 20
  # creamos el video con los datos obtenidos
  video=cv2.VideoWriter(srcVideo, fourcc, fps,(width,height), True)
  for img in images:
    video.write(img)
  cv2.destroyAllWindows()
  video.release()
```

Esta función está en el fichero **video.py** y se encarga de crear el vídeo a partir de las imágenes creadas. Primero se introducen las imágenes de información *srcImg* y las correctoras del ruido *srcImgRuido* en una lista y se calculan las dimensiones de las imágenes.

Después se crea el codec que se utilizará, en este caso *HFYU* y se indican los fps que se desean (*recomendado menos de 100 fps*).

Para acabar se crea el vídeo y se introducen todas las imágenes que se han creado con anterioridad. Los vídeos son guardados en ***proyectoVideoData/VideoDataDB/VideosDB***.

Corrector de ruido

Al crear cada frame de información se creará su frame correspondiente de compensación y para ello se utilizará una lista de 255 elementos (1 por cada valor de píxel posible) que se guarda en un fichero binario llamado correctorRuido.bin.

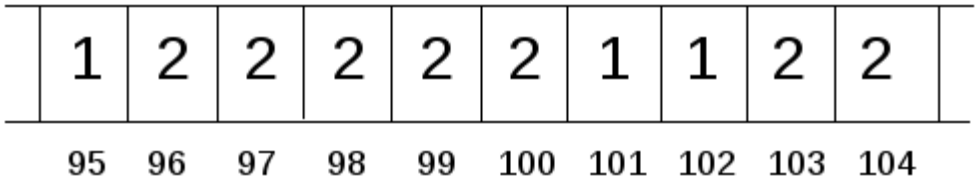


Ilustración 46: Corrector de ruido

El corrector es una lista donde en la posición de la lista 97 guarda un 2 porque es el ruido insertado al guardar un byte con valor 97, de esta forma cuando se guarde un byte con valor 97 se recuperará el byte 95. Esta lista es usada en la creación del vídeo para en función de estos valores crear el frame con la supresión de ruido por cada imagen de información.

Esta lista es creada comparando una imagen que contiene todos los valores posibles (*0-255*) con la misma imagen tras recuperarse del vídeo. De tal forma que se añade a la lista para cada valor la diferencia entre el píxel correspondiente.

La imagen de supresión de ruido que es creada por cada imagen con información contienen en cada píxel un 100 si hay que añadir al píxel recuperado con información un 1 o un 200 si hay que sumar al píxel un 2.

La imagen de supresión por cada imagen de información no sería necesaria si siempre fuese el mismo valor pero en este caso puede ser un 1 o un 2 y es necesario saber cual es el valor concreto para recuperar la información exacta que se guardó.

Indexado del vídeo

Algo que es imprescindible para obtener un rendimiento óptimo y factible en la recuperación del ficheros a partir del vídeo es el uso de índices. El uso de vídeos como base de datos tiene sus ventaja e inconvenientes, por ello su uso debe ser para un ámbito específico debido a que una vez creado el vídeo con los ficheros ya no puede eliminarse un fichero concreto ni añadirse nuevo contenido al vídeo. Claramente el uso de este tipo de vídeos es para almacenamiento de historiales que será grandes cantidades de datos e información fija que no se modificará y solo se utilizará como consulta de datos.

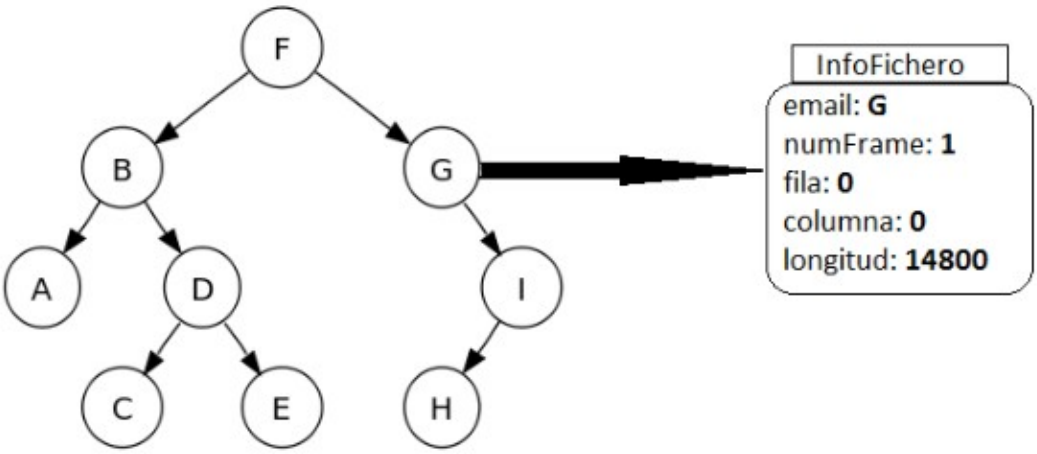


Ilustración 47: Estructura del índice de los VideoDatas

Teniendo en cuenta las características de estos vídeos de almacenamiento, la estructura elegida será el *árbol binario* que estará implementado mediante el fichero **arbolBinario.py** donde cada elemento será un objeto de la clase **InfoFichero** que contiene la información necesaria para obtener el contenido del fichero en el vídeo de una forma óptima.

InfoFichero tiene el *identificador del email* que es utilizado para buscar e insertar en el índice, el *número del frame* donde se encuentra el fichero para acceder de forma directa a dicho frame sin tener que pasar por los anteriores del vídeo, las *coordenadas* donde comienza el fichero dentro del frame y la *longitud* del fichero.

Cuando los índices son creados en la creación del vídeo, se guardan como ficheros binarios en *proyectoVideoData/VideoData/Python/videodata/indicesVideos.* La módulo **indexado.py** se encarga de la creación y recuperación de los índices.

```
def crearIndiceVideo(nombreModelo):
             listaFicheros       =       getFicheros("../../VideoDataDB/Respuestas/
{0}/*{1}".format(nombreModelo, EXTENSION))

  for fich in listaFicheros:
    tam = os.path.getsize(fich)
    pos += tam
    # se crea el elemento a indexar
     examen = infoFich.InfoFichero(obtenerNombreFichero(fich), numFrame, fil, col,
tam)
    # el elemento creado es insertado en el arbol
    abb.agregarElemento(arbol, examen)
    if (TAMTOTAL) < pos:
      incFrames = pos/TAMTOTAL
      pos = pos-(incFrames*TAMTOTAL)
      numFrame += incFrames
      # las coordenadas son recalculadas
      fil, col = obtenerCoordenada(pos)

  # se guarda el arbol en un fichero con el nombre del modelo y extension '.bin'
  abb.guardarArbol(nombreModelo, arbol)
```

El algoritmo de creación del índice consiste en dado el nombre del modelo o actividad, se obtienen todos los ficheros de respuestas de los alumnos y se calcula la posición que ocuparán en el vídeo insertando esta información en el árbol que es usado como índice. Finalmente se guarda el índice en un fichero binario usando el almacenamiento externo para posteriores recuperaciones del índice porque se utilizará en la obtención de ficheros.

Recuperación de ficheros del vídeo

La recuperación de ficheros consiste en recuperar del vídeo un fichero o ficheros concretos, para ello se utiliza el índice que es recuperado, se busca la información del fichero que contiene donde se encuentra dentro del vídeo y se procede a recuperar ese fichero accediendo al frame o frames concretos.

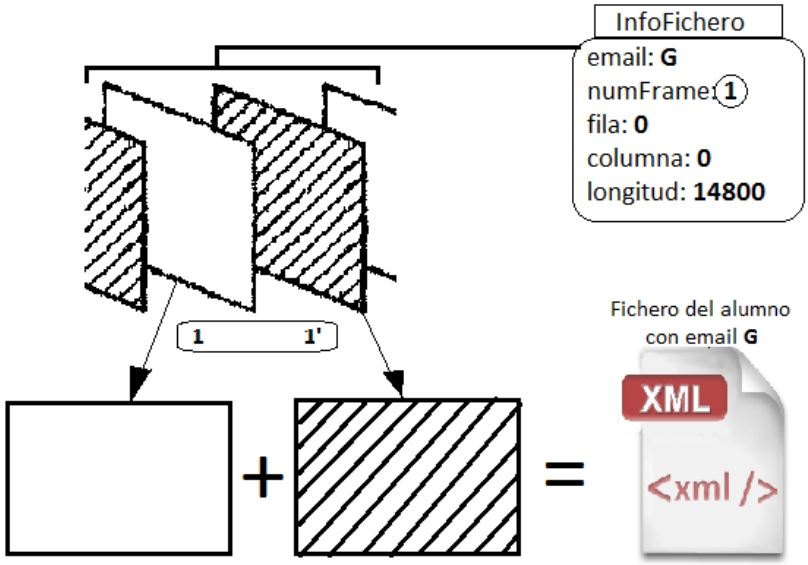


Ilustración 48: Obtener fichero del vídeo

Teniendo la información del fichero el proceso consiste en consultar el numero del frame donde se encuentra, acceder al frame concreto del vídeo y extraer ese frame y el siguiente que contiene el corrector de ruido.

Una vez tenemos los dos frames hay que realizar una reconstrucción que finalmente da como resultado el fichero de alumno con las respuestas y resultados.

```
def obtenerFichero(nombreModelo):
 # se pide el email del fichero a buscar
  emailABuscar = raw_input("Introduce el identificador de email:")
  # recuperamos el indice y buscamos usando el email
  arbol = index.recuperarIndiceVideo(nombreModelo)
  existe, fichUsuario = abb.buscar(arbol, emailABuscar)
  if existe:
    # se extrae la informacion de los frames necesarios para obtener el fichero
    texto = obtenerInfo("../../VideoDataDB/VideosDB",nombreModelo, fichUsuario)
    # creamos el fichero en la ruta determinada y le ponemos
    # como nombre el email y extension xml
    outfile = open('../../VideoDataDB/Recuperado/{0}/
                    {1}.xml'.format(nombreModelo,emailABuscar), 'w')
    outfile.write(texto)
    outfile.close()
```

El algoritmo contenido en el fichero **videodata.py** se encarga de obtener un fichero de la actividad *nombreModelo*; para ello se pide el email del alumno que queremos recuperar, se recupera el índice del vídeo de la actividad, se recupera la información del fichero del email introducido y si existe se obtiene la información del alumno usando el vídeo. Finalmente este texto extraído es guardado en un fichero en ***proyectoVideoData/VideoDataDB/Recuperado/nombreModelo/*** con formato xml.

El método fundamental es *obtenerInfo* que obtiene el contenido del fichero usando el vídeo como base de datos y empleando la información extraída del índice para localizar el frame y posición del fichero en el vídeo.

```
def obtenerInfo(ruta, nombreModelo, infoFichero):
  LIMITE = COLUTIL*FILAS
  tamRestante = infoFichero.longitud
  # obtenemos el numero del frame
  numFrame = infoFichero.numFrame
  # construimos la ruta completa hasta el video
  rutaVideo = "{0}/{1}.avi".format(ruta, nombreModelo)
  # calculo la posicion de inicio del fichero dentro del frame
  pos = (infoFichero.fila*COLUTIL)+infoFichero.columna

  # el proceso no acaba hasta que tengamos todas la informacion
  while tamRestante != 0:
    # obtenemos el frame y despues su informacion
    img = video.reconstruirImg(numFrame, rutaVideo)
    cad = video.getInfo(img)
   # si es el ultimo frame a procesar
    if (pos+tamRestante) <= LIMITE:
      # acumulamos la info desde la pos hasta la pos + tamRestante
      infoUtil += cad[pos:pos+tamRestante]
      # ya tenemos toda la info y el tamRestante es 0
      tamRestante = 0

    # si no, se procesa el frame entero
    else:
      infoUtil += cad[pos:]
      tamRestante -= LIMITE-pos
      pos = 0 # ponemos la pos a 0 para el siguiente frame a procesar
      numFrame += 1 # pasamos al siguiente frame

       # devolvemos la info util
       return infoUtil
```

El algoritmo *obtenerInfo* va accediendo a los frames donde se encuentra el fichero dentro del vídeo y **va reconstruyendo los frames para obtener el contenido** del fichero **hasta que tenga el tamaño indicado en la campo longitud del elemento recuperado del índice**.

Un método muy importante es la reconstrucción del frame que consiste en utilizar el frame que contiene el corrector del ruido para crear el frame tal y como se insertó en un inicio corrigiendo el ruido y obteniendo la información correcta.

```
def reconstruirImg(num, srcVideo='frames/videoBD.avi'):
  # obtenemos el frame de informacion, su correspondiente de ruido y estado
  flag, img, imgRuido = getFrame(srcVideo, num)

  if flag:
    height , width = img.shape[:2]
    # creamos la imagen donde se restaurara
    imgRestaurada = np.zeros((height,width), np.uint8)
    # recorremos la imagen
    for i in range(height):
      for j in range(width):
        p = img[i][j] # copiando cada pixel
       # se observa el ruido en su imagen de correccion y se corrige
        if imgRuido[i][j] < 150:
          p = p + 1
        else :
          p = p + 2
        imgRestaurada[i][j] = p # vamos restaurando la imagen
  return imgRestaurada
```

El algoritmo *reconstruirImg()* forma parte del fichero **video.py** y recibe como parámetros el número del frame a reconstruir y la ruta al vídeo, utiliza el método *getFrame()* que obtiene el frame indicado por parámetros y el siguiente que contiene el corrector del ruido, después se recorre la imagen extraída, se obtiene cada píxel que es un byte y en función de la imagen con el ruido se incrementa el valor en uno o dos debido a que esta imagen de corrector de ruido contiene en cada píxel el valor 100 si hay que incrementar lo recuperado en uno y el valor 200 si hay que incrementar el valor recuperado en dos. Después de reconstruir toda la imagen esta es retornada y ya contiene la información guardada al principio con el contenido exacto.

Una función que utiliza es *getFrame(src,num)* que es fundamental debido a que **obtiene las imágenes indicadas por parámetro de una forma directa mediante acceso aleatorio debido a que el algoritmo no sería eficiente si el acceso al frame del vídeo fuese en acceso secuencial**.

```
def getFrame(dirVideo, num):
 # abrimos el video
  videoCapture = cv2.VideoCapture(dirVideo)

 if videoCapture.isOpened():
    # nos posicionamos en el frame par, que es el que incluye la informacion
    videoCapture.set(cv2.cv.CV_CAP_PROP_POS_FRAMES, num*2)

   # obtenemos el frame de informacion
    retval, image = videoCapture.read()
    if retval:
      # obtenemos el frame de ruido o dispersion del codec
      retval, imageRuido = videoCapture.read()

      imageGray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
      ruidoGray = cv2.cvtColor(imageRuido, cv2.COLOR_BGR2GRAY)
```

```
    # devolvemos el flag que indica que todo ha ido bien y los frames recuperados
    return [retval, imageGray, ruidoGray]
```

La función *getFrame* recibe como parámetros la ruta del vídeo y el número del frame a obtener. El primer paso es abrir el vídeo, después se pone el *puntero* sobre el frame indicado por parámetro multiplicando por dos debido a que por cada frame de información existe un frame de dispersión del codec o eliminación de ruido. Después se obtiene la imagen de información y la imagen de dispersión del codec que se encuentra a continuación del frame de información. Finalmente se pasa a gris debido a que los frames no utilizan RGB sino escala de grises donde cada píxel guarda un byte correspondiente a un carácter.

Capítulo 5.- Manual de usuario

Con este manual el usuario aprenderá a utilizar la aplicación **VideoData** de una forma rápida comprendiendo su uso mediante la interfaz proporcionada.

¿Como crear el contenido?

Antes de empezar a crear el contenido (hipervídeos y actividades) resulta que es imprescindible la fase de planificación donde el profesor debe plantearse cuales son los objetivos de aprendizaje que quiere que adquieran sus alumnos.

Hay que tener en cuenta que **el hipervídeo es un vídeo con una estructura interactiva que utiliza la asociación de distintos tipos de elementos o contenidos durante la secuencia temporal de reproducción** que es lineal. Están enriquecidos permitiendo pausar la secuencia de reproducción para ampliar el conocimiento utilizando otros elementos que forman esa estructura del hipervídeo: notas, imágenes, actividades, etc.

Concretamente entre esos elementos en este sistema, destacan **las actividades** que **permiten mejorar y repasar lo aprendido en el vídeo mediante una serie de preguntas que pueden ser de tipo test (simple o múltiple), desarrollo, recursos, etc**.

El siguiente paso, **una vez que estén claramente especificados los objetivos de aprendizaje, es proceder a la creación de actividades** que se consideren oportunas para reafirmar el conocimiento del temario por parte de los alumnos.

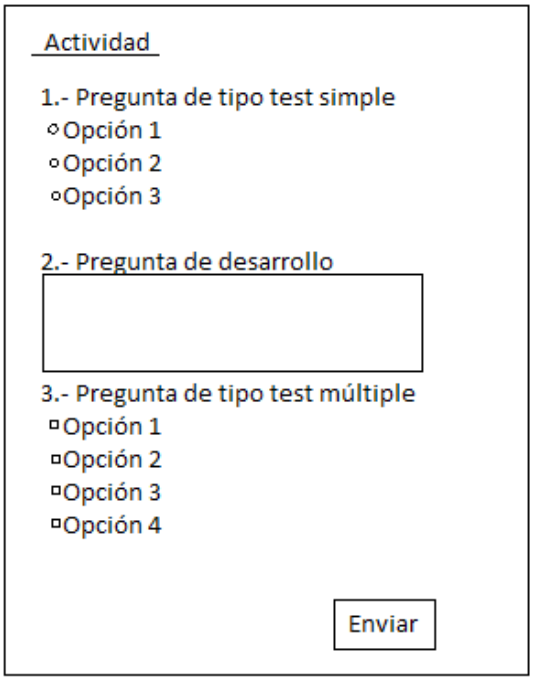


Ilustración 49: Ejemplo de actividad

En la imagen anterior se puede ver un ejemplo de una **actividad que se utilizará para reafirmar los conceptos que se deseen enseñar a los alumnos.** Estas actividades se dispondrán intercaladas con los recursos de aprendizaje que va guiando el vídeo para evaluar sus conocimientos.

Este estudio previo es importante ya que a partir de ahí se puede proceder a la creación del vídeo donde se impartan los conceptos deseados y debe tenerse en cuenta el tiempo en el que comienza cada fase de aprendizaje dentro del vídeo anotando el segundo de inicio e indicando que es lo que se mostrará al alumno de la estructura del hipervídeo (subtítulo, nota, imagen, actividad). Sobre todo **deben especificarse claramente el inicio de cada actividad** que se ha creado con anterioridad y hay que tener en cuenta que **todas las preguntas de la actividad deberían poder ser contestadas correctamente si se ha visualizado la parte correspondiente del vídeo**.

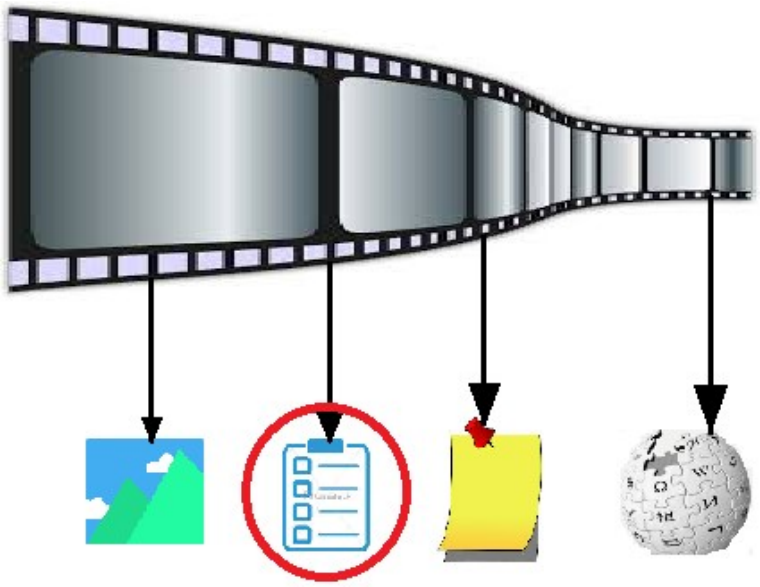


Ilustración 50: Planificación del hipervídeo

Como se muestra en la ilustración anterior, **se procede a la visualización del vídeo creado y se marcarán los puntos donde debe aparecer cada elemento que forma parte de la estructura del hipervídeo**. El elemento de la estructura del hipervídeo señalado en rojo coincide con una actividad de las que se habla con anterioridad que contendrán una serie de preguntas.

Después de tener ya el prototipo completo del vídeo, las actividades y donde va cada actividad ya se puede proceder con la creación de los ficheros de entrada. Esto se puede ver en el capítulo de Entrada de datos.

Especificación de contenidos: Exámenes e Hipervideos

En esta sección se comentará el formato de los ficheros de entrada tanto para la creación de **exámenes** como de **hipervídeos**, en los que enlazaremos a exámenes y otro tipos de actividades o recursos..

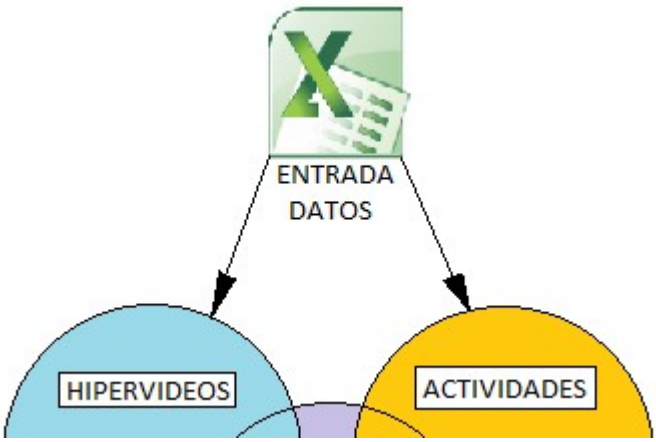


Ilustración 51: Entrada de datos en formato xls

La entrada de datos de los dos tipos de contenido (actividades (de tipo exámenes) e hipervídeos) consisten en hojas de cálculo Excel con formato XLS.

Formato de entrada de actividades

Con el objetivo de evaluar el aprendizaje de los alumnos se ha desarrollado un sistema que permite crear actividades y poder evaluarlas automáticamente si son preguntas de tipo test o de forma manual por parte del profesor si son preguntas de desarrollo o recursos.

Antes de comprender el formato de entrada de actividades hay que saber como se representarán los elementos una vez creados para tener una visión previa. Por ello las actividades contienen los siguientes tipos de preguntas.

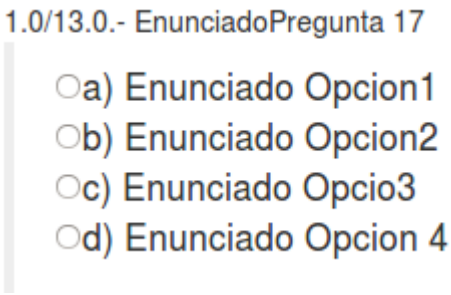


Ilustración 52: Pregunta test simple

En las preguntas de tipo test simple solo se puede seleccionar una respuesta posible y es obligatorio la selección de una de ellas.

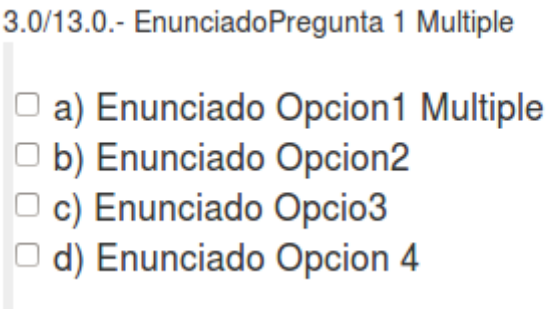


Ilustración 53: Pregunta test múltiple

Las preguntas de test múltiples permite la opción de seleccionar varias respuestas. También se puede dejar sin seleccionar ninguna opción.

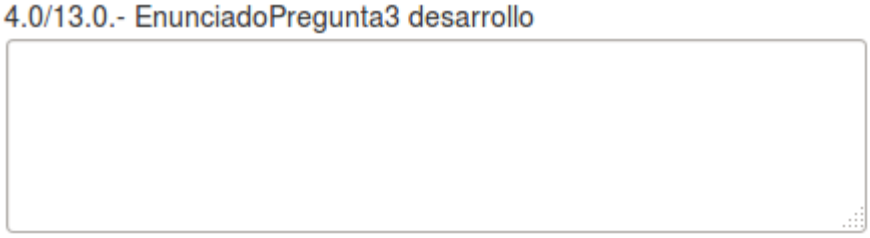


Ilustración 54: Pregunta de desarrollo

Las preguntas de desarrollo contiene el campo para introducir teoría respondiendo a la pregunta. El campo debe rellenarse con algún texto.

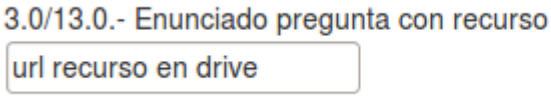


Ilustración 55: Pregunta de tipo recurso

Las preguntas de tipo recurso contienen el campo para introducir una url hacia el recurso creado para responder a la pregunta. El recurso es conveniente que se encuentre en la nube, pudiendo usar Google Drive o Dropbox. Es un campo obligatorio y solo se puede especificar un recurso, si hay varios recursos es conveniente que los organice todos en una carpeta de Drive o Dropbox e introducir el enlace a la carpeta.

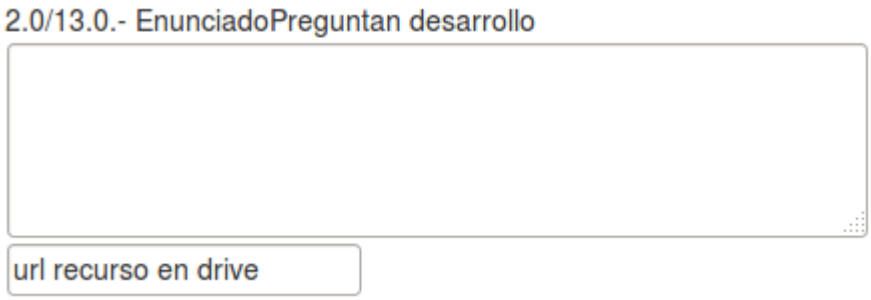


Ilustración 56: Pregunta de tipo DesRec

Es la mezcla de las dos preguntas anteriores, contiene un campo para desarrollar la respuesta que será de tipo teoría y otro campo para introducir la dirección al recurso creado para tener bien la pregunta. Ambos campos son obligatorios.

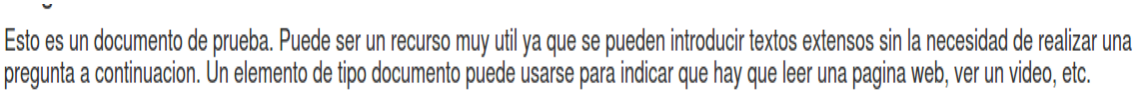


Ilustración 57: Elemento de tipo Documento

Por último el elemento documento consiste en mostrar el texto indicado, es un recurso muy útil ya que se pueden introducir textos extensos sin la necesidad de realizar una pregunta a continuación. Un elemento de tipo documento puede usarse para indicar que hay que leer una página web, ver un vídeo, etc.

En este caso utiliza un fichero con una estructura determinada que se detallará a continuación y **opcionalmente otro fichero con metadatos a incrustar en la actividad creada**. **Estos ficheros para el correcto funcionamiento deben estar en**

el directorio *proyectoVideoData/EntradaDatos/Actividad*.

	A	B	C	D	E	F	G	H	I	J	
1	Tag	Puntos	Pregunta			Tipo	Solucion	NumRespuestas	Enunciados Opciones		
2	tagPregunta2	1	EnunciadoPregunta 17			TestSimple	3	4	Enunciado Opcion1	Enunciado Opcion2	Enu
3	tagPregunta	3	EnunciadoPregunta 1 Multiple			TestMultiple	1,3	4	Enunciado Opcion1	Enunciado Opcion2	Enu
4	tagDesarrollo	4	EnunciadoPregunta3 desarrollo			Desarrollo	Respuesta correcta				
5	tagPregunta	3	Enunciado pregunta con recurso			Recurso	urlSolucion				
6	desarrollo	2	EnunciadoPreguntan desarrollo			DesRec	Solucion Desarrollo | urlSolucionRecurso				
7	tagPregunta2	2	Texto de un documento			Documento	Solucion				

Ilustración 58: Formato de entrada de actividades

En la ilustración anterior se puede ver el formato del fichero de entrada para crear las actividades. Los detalles a tener en cuenta son:

- La primera línea es de información con los nombres de las columnas que se usan en la hoja de cálculo.
- En la primera columna se puede indicar un tag para permitir filtrar las preguntas por el temario al que pertenezca.
- La segunda columna contiene los puntos que vale dicha pregunta. Si el campo está vacío el valor de la pregunta será de 0.
- La tercera columna contiene los enunciados de las preguntas o el texto del documento si se trata de este tipo.
- Después hay dos columnas vacías.
- La columna de tipo debe contener el tipo concreto que se desee. Estando los siguientes tipos:
    - **TestSimple:** es una pregunta de tipo test donde solo una de las opciones son correctas. Si el alumno selecciona otra opción se le restará la mitad del valor de esa pregunta.
    - **TestMultiple:** es una pregunta de tipo test que puede tener múltiples opciones correctas. Para evaluar esta pregunta se resta un tercio del valor de una opción correcta por cada respuesta errónea.
    - **Desarrollo**: son preguntas donde la respuesta es el desarrollo de un texto por parte del alumno. Pudiendo usarse como preguntas de teoría.
    - **Recurso:** son preguntas donde se pide algo práctico (ej: diagramas, planos, etc) y la respuesta consiste en indicar la url de la dirección

donde se encuentra ese recurso. Se recomienda usar Google Drive o Dropbox.

- **DesRec:** es la mezcla de los dos tipos anteriores. Consiste en pedir una parte teórica en el desarrollo y otra práctica para el recurso.
- **Documento:** es un texto que el alumno debe leer.

- En la columna de solución se describe la solución ideal a la pregunta, esta va en función del tipo de pregunta. La solución indicada debe ser de la siguiente forma en función del tipo:
  - TestSimple: debe tener el número de la opción correcta, siendo 1 para la primera opción y los números correspondientes para las siguientes.
  - TestMultiple: debe tener un conjunto de números separados por coma que corresponden al número de opción correcta. Ej: *si hay un test multiple con 4 opciones y en el campo solución aparece '1,2', significa que las dos primeras opciones son las correctas.*
  - Desarrollo: el texto de teoría idóneo que debe introducir el alumno.
  - Recurso: el enlace a una práctica realizada que responda a la respuesta indicada.
  - DesRec: contiene la teoría separada por '|' como separador del enlace de la solución del recurso. Ej: '*solucion de la parte del desarrollo*' | *urlRecurso.*
- Las siguientes columnas solo se utilizan para las preguntas de tipo test, siendo una para el número de opciones que tiene la pregunta de tipo test y el resto de columnas para indicar las opciones disponibles.

	A	B
1	CLAVE	VALOR
2	Autor	Jorge Casiano Bermejo
3	Asignatura	PIM
4	Idioma	ES

Ilustración 59: Metadatos de la actividad

La entrada de metadatos se realiza usando otro fichero aparte Excel en formato xls. Este fichero no es obligatorio pero si se quieren introducir metadatos hay que utilizar el formato que aparece en la imagen de arriba. Donde la primera línea es informativa

y las siguientes son los metadatos usando la primera columna como clave y la segunda como valor.

La única restricción que tiene este fichero es que no se deben introducir metadatos cuya clave se llame 'titulo' o 'nombre' debido a que son unos metadatos obligatorios que se introducen por teclado en el proceso de creación de actividades.

Puede usarse como plantilla el fichero *BateriaPreguntasTabla.xls* que se encuentra en el directorio *proyectoVideoData/EntradaDatos/Actividades* y también se encuentra en *proyectoVideoData/VideoDataDB/test/entradaDatos/actividades*. Este fichero contiene todos los elementos que puede tener un examen y puede ser usado como plantilla.

Formato de entrada de hipervídeos

En la creación de contenido para impartir el temario se utilizan hipervideos que consisten en la utilización de vídeos en el aprendizaje incorporando eventos a este para mejorar la experiencia de los alumnos e incrementar la capacidad de aprendizaje extendiendo el uso de un simple vídeo.

En la creación de estos hipervídeos se utilizan como entrada dos ficheros de tipo Excel con formato XLS:

1. **Uno con los eventos o capas que se crearán sobre el vídeo que llamaremos estructura del vídeo.**
2. **Otro con los metadatos que contienen la información sobre el vídeo** a reproducir, identificador, tipo de vídeo, duración, etc.

Estos ficheros Excel que serán usado como entrada de datos para la creación de los hipervídeos deben estar en el directorio *proyectoVideoData/EntradaDatos/Popcornjs*.

Puede probarse el fichero *proyectoVideoData/VideoPopcorn/hipervideo DePruebaInterno.html* para ver un ejemplo de hipervídeo.

Estructura del hipervídeo

Los eventos o estructura del vídeo usados están claramente ligados al vídeo y se

activan y desactivan **en función del tiempo de reproducción del vídeo**.

Una descripción de las capas o eventos que se pueden insertar en los vídeos formando la estructura para la creación de estos hipervídeos:

- **Índice**: permite moverte a un punto concreto del vídeo sabiendo que se va a encontrar el alumno. Su uso es exactamente igual que los índices de un libro facilitando y mejorando la navegación.
- **Escenas**: mediante la creación de escenas podemos manipular el flujo de reproducción del vídeo mostrando al alumno las escenas que quiera el profesor y que no tenga que ver trozos del vídeo donde se enseña algo que este alumno ya ha aprendido.
- **Subtítulos**: tanto internos como en un fichero con formato srt, con esta función podemos reducir la limitación del idioma.
- **Notas:** se pueden añadir notas a los vídeos donde se realicen aclaraciones o se extienda el contenido de una forma más detallada.
- **Imágenes:** es posible añadir imágenes que aparecerán en el navegador mientras se reproduce el vídeo.
- **Mapas:** se pueden poner mapas para enriquecer el aprendizaje.
- **Web:** es posible añadir páginas web y **también las actividades que haya creado el profesor** para que sea realizada por los alumnos.
- Feed: se pueden indicar ficheros RSS también que pueden contener noticias.

	A	B	C	D	E	F	G	H
1	VIDEOS		videoEjemplo					
2	TIEMPO		0	5	8	10	15	30
3	SUBTITULOS		textoSub1	0			textoSub2	
4	NOTAS		textoNota				textoNota2	0
5	IMAGENES		imagen.png		0			imagen2.png
6	MAPAS		1,England,ROADMAP					
7	WEB			http://localhost:8080/VideoDataWeb/actividadConPreguntasVariadas.html				
8	FEED						urlFeed	
9	ESCENAS		inicioEscena1			finEscena1	inicioEscena2	
10	INDEX		Introduccion		Presentacion		Habla un personaje	

Ilustración 60: Formato hipervídeo

Este es el formato utilizado en la creación de los hipervídeos. Este es un fichero que se proporciona como prototipo llamado **videoEjemplo.xls.**

En los eventos SUBTITULOS, NOTAS, IMAGENES, MAPAS, WEB y FEED se indica el final del evento concreto con un 0 o si no se terminará el evento en el fin de

esa escena. Ej: *la imagen 'imagen.png' aparece en el segundo 0 y desaparece en el segundo 8, en cambio la nota 'textoNota' aparece en el segundo 0 y desaparece en el segundo 10 ya que es el final de esa escena*.

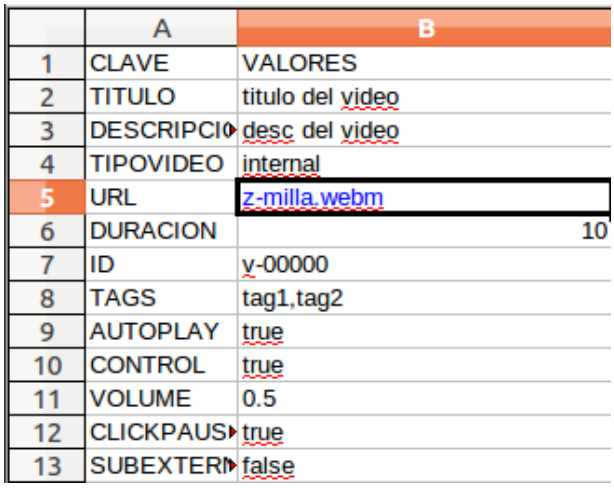
En la imagen anterior se puede apreciar el formato utilizado para crear un hipervídeo, donde hay que seguir para la creación los siguientes consejos:

- **La segunda columna debe estar vacía** por cuestiones de formato.
- La primera línea contiene el nombre del vídeo aunque no es una información funcional debido a que este dato se obtiene de la entrada de metadatos.
- **TIEMPO contiene los tiempos donde empiezan o terminan los eventos**.
- **SUBTITULOS** pueden ser usados desde dos tipos de recursos:
    - **Internos**: son los que se indican en esta entrada de datos, consisten en indicar el texto de subtítulo en el segundo concreto donde aparecerá en el vídeo. Ese texto como subtítulo desaparecerá en el segundo donde haya otro texto, un 0 o sea el fin de una escena. **Es imprescindible para utilizar los subtítulos internos tener el campo correspondiente de la entrada de metadatos a 'false'**.
    - **Externos**: los subtítulos son cargados desde un fichero externo, se activan desde la entrada de metadatos indicando el nombre del fichero. Utiliza el formato srt y estos ficheros se deben encontrar en el directorio ***proyectoVideoData/VideoDataWeb/WebContent/video*/sub** esta opción es explicada con más detalle a continuación en los metadatos del hipervídeo. **Si se utiliza esta opción, los subtítulos de la entrada de la estructura del hipervídeo no será tenido en cuenta.**
- **NOTAS** es texto mostrado mientras se está reproduciendo el vídeo. Puede usarse para aportar información extra en relación al contenido en cierto momento del vídeo.
- **IMAGENES** contiene las imágenes que se mostrarán durante la reproducción del hipervídeo. Debe contener el nombre de la imágencon el formato incluido (*Ej: imagen.**png***) y deben estar en el directorio **proyectoVideoData/VideoDataWeb/WebContent/video/img** para el correcto funcionamiento.

- **MAPAS** muestra el mapa de una localización determinada durante la reproducción del vídeo. El formato de los mapas es el zoom que es un número (que indica el zoom del mapa), la localización y el tipo de mapa permitiendo (ROADMAP, SATELLITE, TERRAIN, STREETVIEW), separados por comas. Ej: *'2,England,SATELLITE' significa el mapa de tipo satélite de Reino Unido con el zoom x2*.
- **WEB** es utilizado para indicar las páginas webs que el usuario debe visualizar en un momento concreto del vídeo. **Es un campo muy importante porque es usado para incluir los actividades diseñadas por el profesor** como aparece en la ilustración con la actividad o examen *http://localhost:8080/VideoDataWeb/actividadConPreguntasVariadas.html*.
- **FEED** contiene páginas que **utilizan servicios RSS** para actualizar los contenidos en tiempo real. Son muy utilizados en blogs y webs de noticias.
- **ESCENAS** debe contener entre el inicio de la primera escena y el inicio de la segunda escena, el final de la primera escena. Ya que para empezar una nueva escena primero debe acabar la anterior.
- **INDEX** contiene los puntos concretos por los cuales los alumnos podrán desplazarse por el vídeo. Su uso es exactamente igual que los índices de un libro facilitando y mejorando la navegación.

Metadatos del hipervídeo

La hoja de cálculo de metadatos debe tener el siguiente formato.



	A	B
1	CLAVE	VALORES
2	TITULO	titulo del video
3	DESCRIPCIO	desc del video
4	TIPOVIDEO	internal
5	URL	z-milla.webm
6	DURACION	10
7	ID	v-00000
8	TAGS	tag1,tag2
9	AUTOPLAY	true
10	CONTROL	true
11	VOLUME	0.5
12	CLICKPAUS	true
13	SUBEXTERI	false

Ilustración 61: Metadatos hipervideo

Como se puede apreciar en la imagen anterior, desde este fichero excel con metadatos se configura el hipervídeo indicando los campos que aparecen en él. Es el fichero **metaVideoInterno.xls** que puede ser usado como prototipo. Una pequeña descripción es la siguiente:

- **La primera columna debe contener de forma obligatoria esas claves y en ese orden.**
- TITULO contiene el título del hipervídeo.
- DESCRIPCION es un texto que describirá el vídeo.
- TIPOVIDEO indica el tipo de vídeo que se permite crear, los valores disponibles deben ser http, internal o vimeo. NOTA: el vídeo de tipo vimeo no permite crear hipervideos así que no es recomendable su uso.
- URL: si TIPOVIDEO es http o vimeo el valor debe ser su url, si por el contrario el tipo es **internal** el valor de URL debe ser el nombre de un fichero de tipo vídeo con el formato incluido donde este fichero debe estar en el siguiente directorio *proyectoVideoData/VideoDataWeb/WebContent/video/video*.
- DURACION: debe contener la duración del vídeo siendo un número en segundos. NOTA: no es trascendental, ya que es meramente informativo.
- ID: es el identificador del vídeo, tiene un formato concreto que es 'v-'

seguido de cinco números. Ej: *v-12345.*

- TAGS: contiene los tags del vídeo, es informativo.
- AUTOPLAY: su valor debe ser 'true' o 'false' y es usado para indicar si al cargar la página se reproducirá de forma automática el vídeo o por el contrario el alumno deberá pulsar el play.
- CONTROL: su valor debe ser 'true' o 'false' y es usado para indicar si se quiere que se muestren los controles sobre el vídeo.
- VOLUME: debe ser un número entre 0-1 e indica el nivel del volumen del audio del vídeo.
- CLICKPAUSE: indica si al pulsar sobre cualquier enlace se pausará el vídeo, su valor debe ser 'true' o 'false'. Es recomendable que el valor sea 'true' ya que al realizar alguna actividad es conveniente que se pare el vídeo.
- **SUBEXTERNAL: su valor es 'false' si no utiliza subtitulos externos en cuyo caso usará los indicados en el Excel de la Ilustración 21 o si se quiere utilizar un fichero de subtítulos externos el valor sera 'nombreFichero.srt' estando este fichero en el directorio *proyectoVideoData/VideoDataWeb/WebContent/video*/sub. NOTA: si se utilizan subtítulos externos no se utilizarán los subtítulos de la Ilustración 21 y el formato de dicho fichero debe ser srt.**

Se puede utilizar como plantilla el fichero *videoEjemplo.xls* que contiene la estructura del hipervídeo y el fichero *metaVideoHTTP.xls* que contiene los metadatos del hipervídeo. Ambos ficheros se encuentran en *proyectoVideoData/EntradadDatos/Popcornjs* y también en el directorio *proyectoVideoData/VideoDataDB/test/entradaDatos/popcorn.*

Estructura del proyecto y configuración

En esta sección se explicara las cosas que el usuario debe conocer sobre la estructura del proyecto y las configuraciones que debe realizar, pero no se comentará como se ejecuta la aplicación ya que esto se explicará detalladamente en la sección **Ejemplo de uso**.

Es necesario realizar unas pequeñas configuraciones sobre unos ficheros llamados **configuracionRuta.txt** que debe contener la ruta hasta el directorio principal que es VideoData (*ej: /directorio/proyectoVideoData/VideoData*) y **este fichero debe estar** en el propio **directorio VideoData,** en el directorio **VideoDataWeb** y en el **directorio donde se encuentre eclipse**. Este fichero con la ruta a la carpeta principal **es usado como PATH para localizar la ruta del sistema**.

Si no se utiliza el servidor Tomcat desde eclipse hay que poner el fichero **configuracionRuta.txt** también **en el directorio *Tomcat/bin*** y el servidor debe arrancarse desde ese directorio.

La estructura del proyecto que debemos conocer para el correcto funcionamiento es la siguiente:

- **EntradaDatos:** contiene dos carpetas (Actividades y Popcornjs) donde se incluyen los ficheros de entrada para crear actividades e hipervideos, usando el ficheros Excel con formato **xls**. La estructura de cada Excel será explicado en las posteriores secciones.
- **VideoData**: es **el directorio principal**, hay que configurar el fichero **configuracionRuta.txt** incluyendo la ruta hasta este directorio (*ej: /directorio/proyectoVideoData/VideoData*), contiene la aplicación de creación de actividades e hipervideo llamado **VideoData.jar**. Hay otro fichero ejecutable llamado **VideoDataAuto.jar** que permite realizar las mismas funciones que el ejecutable anterior pero de forma automática mediante la carga de un fichero para automatizar el proceso. El directorio también contiene:
    - **Python**: este directorio contiene lo necesario para guardar los datos en vídeos y recuperar la información a través de estos. La aplicación está contenida en el fichero **main.py**.

- **VideoDataWeb**: contiene la aplicación web que utiliza las actividades creadas para que puedan ser utilizadas por los alumnos. Debe tener otro fichero **configuracionRuta.txt** con la ruta hasta el directorio principal (*ej: /directorio/proyectoVideoData/VideoData*)
- **VideoDataWeb/WebContent/video:** tiene los hipervídeos que contiene la página web e infraestructura para visualizar en el navegador de forma correcta los hipervídeos.
- **VideoPopcorn:** copia de seguridad de los hipervídeos que contiene la página web e infraestructura para visualizar en el navegador de forma correcta los hipervídeos.
- **VideoDataDB:** es el directorio donde se almacenan todos los datos relacionados con las actividades y los resultados de los alumnos. Contiene las siguientes carpetas:
    - **Correcciones:** se almacena un fichero Excel por cada actividad con las preguntas que no pueden ser corregidas de forma automática y los alumnos para corregir la actividad de forma manual a todos los alumnos.
    - **Respuestas:** contiene un fichero Excel por cada actividad con las respuestas de todos los alumnos y una carpeta por cada actividad con las respuestas de cada alumno en un fichero xml que será el contenido que se guarde en los vídeos.
    - **VideoDB:** contiene un vídeo por cada actividad con los resultados y respuestas contestadas de todos los alumnos que han sido guardadas y codificadas en el vídeo.
    - **Recuperado:** almacena en una carpeta por cada actividad los ficheros recuperados a partir del vídeo.
    - *test: contiene una estructura similar a la descrita con ficheros de entrada y sus ficheros de salida correspondientes.*

Ejemplo de uso del sistema VideoData

En esta parte se realizará un ejemplo de uso de cada funcionalidad que se puede hacer desde el sistema completo. El manejo del sistema se realiza mediante línea de comandos.

Si no se usa el servidor desde Eclipse, se arrancará el servidor de forma directa usando la línea de comando.

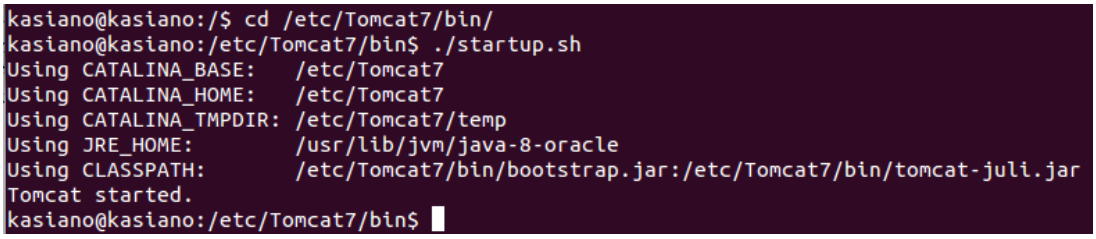


Ilustración 62: Arrancar tomcat desde consola

Ya estaría Tomcat arrancado y para que funcione correctamente el fichero WAR debería estar en el directorio *Tomcat/webapps* y en el directorio *Tomcat/bin* debe estar el fichero **configuracionRuta.txt** con la ruta al proyecto principal (ej: */directorio/proyectoVideoData/VideoData*).

NOTA: Si se usa esta opción, cada vez que se cree un hipervídeo o se inserte una actividad hay que seguir los pasos indicados en Desplegar en servidor (sin eclipse), compilando la aplicación web VideoDataWeb en un fichero WAR e insertarla en *Tomcat/webapps* y volver a desplegar el servidor (./startup.sh).

Si por el contrario usamos Eclipse, lo primero que haremos será arrancar Eclipse que debe estar configurado con el servidor Tomcat y la aplicación web VideoDataWeb desplegada y procedemos a arrancar el servidor.

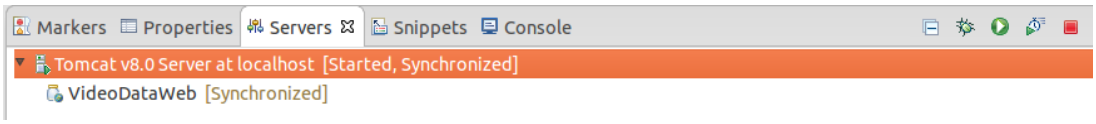


Ilustración 63: Servidor arrancado con VideoDataWeb desplegado

Como se muestra en la imagen anterior, el servidor ha sido arrancado y la aplicación VideoDataWeb ha sido desplegada. La forma más simple de comprobar que está correctamente desplegada es usando el navegador e introduciendo *http://localhost:8080/VideoDataWeb/*.

Veremos el funcionamiento de todo el sistema, para ello usaremos cada funcionalidad disponible. La aplicación esta separada en dos módulos distintos, uno para crear actividades, corregirlas y crear hipervideos y el otro para guardar los datos en un vídeo, recuperar resultados, etc usando un vídeo como base de datos.

Primero se mostrará la funcionalidad del sistema de creación mostrando cada funcionalidad y comentando las posibles opciones y los datos de entrada y salida.

Después se procederá con la ejecución del módulo de almacenamiento y recuperación a través de vídeos.

Tanto en la creación de contenido como en la evaluación de las actividades hay dos opciones de ejecución, una automática donde se utiliza un fichero txt configurado para realizar la función concreta y otra opción manual.

Arranque aplicación de creación

Arranque automático (VideoDataAuto.jar)

```
kasiano@kasiano:~$ cd Escritorio/proyectoVideoData/VideoData
kasiano@kasiano:~/Escritorio/proyectoVideoData/VideoData$ java -jar VideoDataAuto.jar -help
-ocpion nombreFichero.txt
Opciones: -a (crear actividades)
Opciones: -e (evaluar actividades)
Opciones: -h (crear hipervideo)
kasiano@kasiano:~/Escritorio/proyectoVideoData/VideoData$
```

Ilustración 64: Arranque de la aplicación de creación automática

Se utiliza el ejecutable **VideoDataAuto.jar** seguido de una opción de las indicadas y el nombre de un fichero que contiene toda la información necesaria para la función indicada.

Arranque manual (VideoData.jar)

```
kasiano@kasiano:~$ cd Escritorio/proyectoVideoData/VideoData
kasiano@kasiano:~/Escritorio/proyectoVideoData/VideoData$ java -jar VideoData.jar
---------------------------------------
1.- Generar actividad
2.- Corregir actividad
3.- Crear hipervideo
0.- Salir
---------------------------------------
Introduce opcion:
```

Ilustración 65: Arranque de la aplicación de creación manual

Como se observa el la ilustración anterior, se ha ejecutado el sistema de creación que se encuentra en el directorio *proyectoVideoData/VideoData* y se ejecuta el fichero

VideoData.jar mediante el comando *java -jar VideoData.jar*.

Es una interfaz muy simple ya que se puede ejecutar cada funcionalidad introduciendo un número, esta consola se muestra después de ejecutar una función hasta que se seleccione la opción de salir.

Generar actividad (exámenes)

Generar actividades automáticas

Se utiliza el ejecutable **VideoDataAuto.jar** en vez de VideoData.jar para poder generar actividades de forma automática.

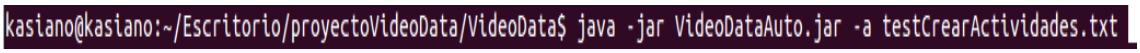


Ilustración 66: Generar actividades automáticas

Para generar las actividades de forma automática hay que elegir la opción **-a** e indicar el nombre del fichero que contiene las actividades a generar. El fichero contiene en cada línea una actividad a generar, usando el siguiente formato.

-e entrada.xls [-m metadatos.xls] -n nombre -t titulo [-tags tag1 tag2 …]

Siendo entrada.xls el nombre de entrada y los campos que están entre corchetes son opcionales. De esta forma la aplicación leerá el fichero y creará una actividad por cada linea con el formato indicado.

Se proporciona el fichero testCrearActividades.txt como ejemplo.

Generar actividad manual

Una vez realizado el arranque manual si escribimos un '1' en la terminal, procederemos a generar una actividad. Para realizar esta función es necesario tener alguna hoja de cálculo como entrada de datos en el directorio *proyectoVideoData/EntradaDatos/Actividad* que sea del formato explicado el la sección **Formato de entrada de actividades**.

Se utilizará el fichero preguntasVariadas.xls como entrada de datos, fichero que es proporcionado como prototipo para crear actividades.

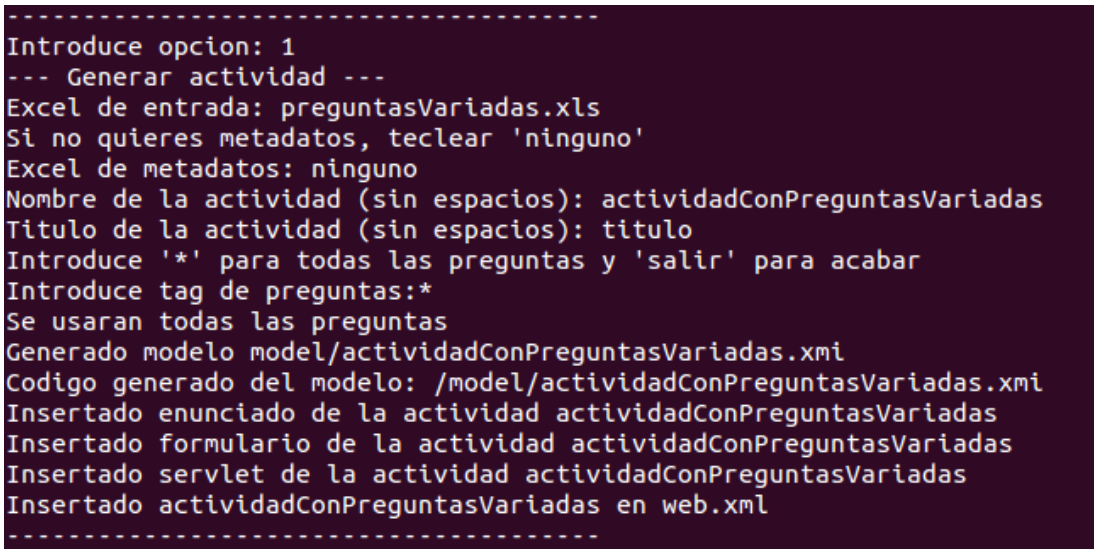


Ilustración 67: Generar una actividad manual

La imagen anterior muestra el proceso completo de generación de una actividad. Después de seleccionar la opción 1, se pide por teclado el excel de entrada que en este caso es un fichero proporcionado para probar el sistema y que coincide con la Ilustración 19.

Una vez introducido el nombre del fichero *preguntasVariadas.xls* que debe ser de un formato especificado y estar en el directorio indicado anteriormente, nos da la opción de introducir el fichero con metadatos pero no es obligatorio en la creación de actividades.

También pide el nombre, en este caso se ha introducido *actividadConPreguntasVariadas* que debe ser un nombre sin espacios como se especifica. Después nos pide introducir los tags que queremos que utilice el sistema para filtrar y seleccionar las preguntas pero se eligen todas las preguntas. Y finalmente nos pide el título de la actividad y se acaba creando esta.

De esta forma generará el código necesario y lo introducirá en la aplicación web VideoDataWeb.
Podemos comprobar que se ha creado la actividad en el directorio *VideoData/model* donde debe existir un fichero con el nombre indicado y formato xmi.

Otra forma para comprobar que está la actividad en la aplicación es abriendo Eclipse, pulsar en la parte de Project Explorer sobre VideoDataWeb, pulsar F5 para actualizar el proyecto, abrir la carpeta del proyecto y la carpeta WebContent y

aparecerá el fichero **actividadConPreguntasVariadas.html**.

Si está el servidor arrancado podemos probar a buscar en el navegador introduciendo *http://localhost:8080/VideoDataWeb/actividadConPreguntasVariadas.html*. Ahora ya estaría listo para usarse esa actividad por parte de los usuarios.

Crear vídeo popcorn (hipervídeo)

Crear vídeos popcorn automáticos

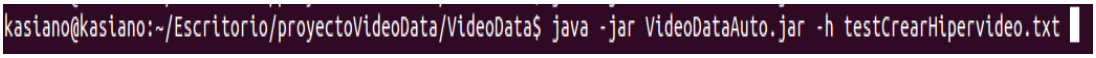


Ilustración 68: Creación de hipervídeos automáticas

Para crear hipervídeos de forma automática hay que elegir la opción **-h** e indicar el nombre del fichero que contiene los hipervídeos a generar. El fichero contiene en cada línea un hipervídeo a crear, usando el siguiente formato.

-e entrada.xls -m metadatos.xls -n nombreVideo

Siendo entrada.xls el nombre de entrada, metadatos.xls los metadatos y la palabra después de **-n** el nombre del vídeo. De esta forma la aplicación leerá el fichero y creará una hipervídeo por cada linea con el formato indicado.

Se proporciona el fichero testCrearHipervideo.txt como ejemplo.

Crear vídeo popcorn manual

Para crear los hipervídeos hay que seleccionar la opción 4 en la consola.

Ilustración 69: Creación de hipervídeo manual

Para la creación del hipervídeo se han utilizado los ficheros indicados como prototipos en la Ilustración 21 para crear el hipervídeo e Ilustración 22 para crear los metadatos. Después se proporciona el nombre del video **videoDePrueba**.

Esto crea un fichero llamado **videoDePrueba.html** en el directorio *proyectoVideoData/VideoDataWeb/WebContent/video que se puede visualizar en un*

navegador que soporte html5 y en el directorio que es usado como copia de seguridad *proyectoVideoData/VideoPopcorn*.

Si se observa la Ilustración 21, se puede ver como en la capa WEB en el segundo 5 se ha insertado la actividad que acabamos de crear *http://localhost:8080/VideoDataWeb/actividadConPreguntasVariadas.html*.

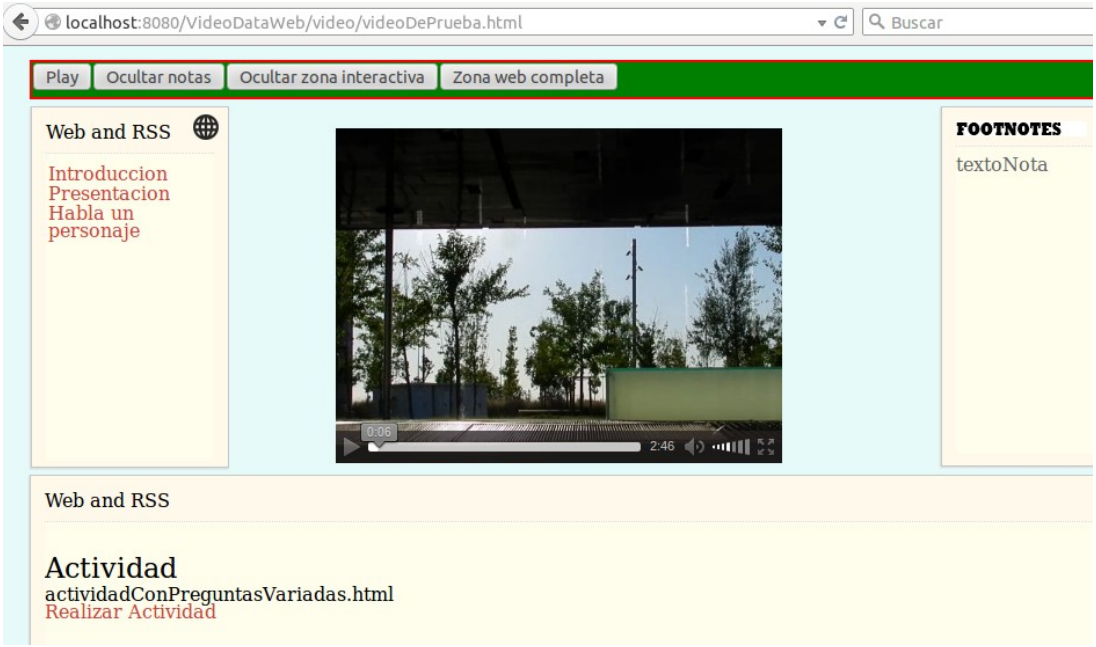


Ilustración 70: Visualización de hipervídeo

Como se puede ver en la imagen el vídeo se para en el segundo seis, uno después de que se muestre la actividad a realizar. Para ello se pulsará sobre **Realizar Actividad** y se abrirá una nueva pestaña con la actividad que hemos creado e insertado en las secciones anteriores.

El servidor debe estar arrancado debido a que la dirección para acceder a los hipervídeos es a través de la aplicación web mediante el servidor y su ruta es ***http://localhost:8080/VideoDataWeb/video/videoDePrueba.html***.

Realizar actividad

Una vez creada e insertada la actividad, ya está disponible en la aplicación web VideoDataWeb. Como el nombre introducido en la creación de la actividad ha sido actividadConPreguntasVariadas la localización de esa actividad en la aplicación es http://localhost:8080/VideoDataWeb/actividadConPreguntasVariadas.html.

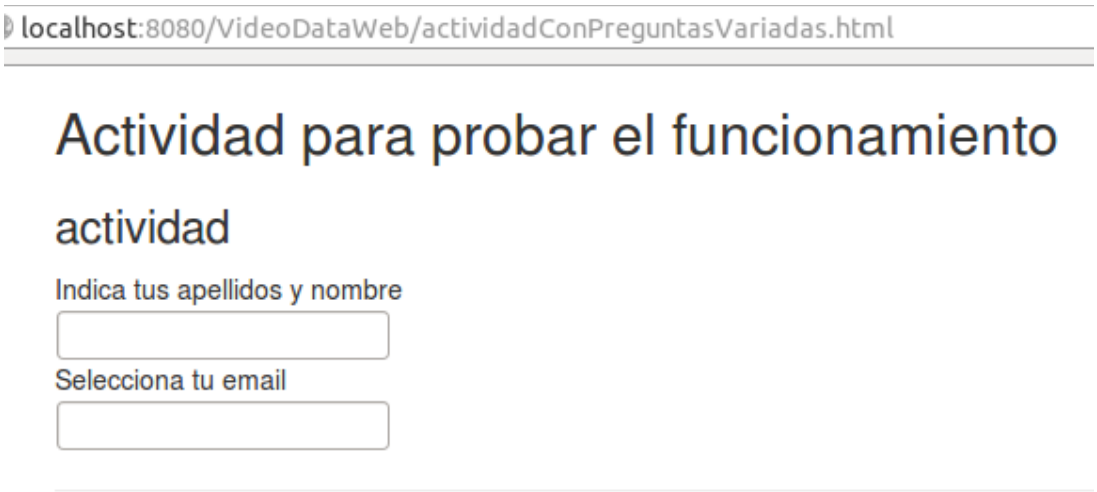


Ilustración 71: Campos obligatorios de la actividad

Como se puede ver en la imagen anterior, la actividad contiene unos campos obligatorios como son el nombre y el email que son usados para identificar cada alumno. Y el resto de preguntas son las recibidas como entrada de datos desde el fichero preguntasVariadas.xls.

Para simular la realización de la actividad por parte de los alumnos hemos rellenado la actividad dos veces usando nombres y emails distintos para ver los datos de salida y los ficheros creados. Los usuarios tienen los emails: pepe@alumnos.unex.es y jcasiano@alumnos.unex.es.

A partir de aquí todos los ficheros con los que se trabaja se encuentran en el directorio proyectoVideoData/VideoDataDB.

	A	B	C	D	E	F	G	H	I
1	Indica tus apellidos	Selecciona tu ema	Enunciado	Enunciado	Enunciado	Enunciado	Enunciado	Preguntan desarrollo	
2	Jorge Casiano Ber	jcasiano@alumno	a) Enuncia	a) Enuncia	Desarrollo	url1	Desarrollo11 | url11		
3	Pepe Santano Parr	pepe@alumnos.ur	b) Enuncia	b) Enuncia	Desarrollo	url pepe 1	Desarrollo pepe 2 | url	pepe 2	
4									

Ilustración 72: Respuestas a la actividad

Todas las respuestas son guardadas en un fichero Excel con el nombre de la actividad en el directorio VideoDataDB/Respuestas. Desde este fichero se pueden ver de forma completa las respuestas de cada usuario a cada pregunta.

jcasiano.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<examen>
  <metadatos>
    <datosPersonales nombreyapellidos="Jorge Casiano Bermejo" email="jcasiano@alumnos.unex.es" />
    <resultados puntuacionMax="13.0" puntuacionMin="6.5" puntuacionObtenida="1.0" notaTotal="7.692308" aprobadoTotal="false"
puntuacionDesRecMax="9.0" puntuacionDesRecObtenida="0.0" puntuacionTestMax="4.0" puntuacionTestObtenida="1.0" notaTest="25.0"
aprobadoTest="false" />
  </metadatos>
  <secciones>
    <seccion nombre="Preguntas">
      <pregunta id="testSimple0" tipo="testSimple" valor="1.0" vObtenido="-0.5">
        <opcion id="a" esCorrecta="false">a) Enunciado Opcion1</opcion>
      </pregunta>
      <pregunta id="testMultiple0" tipo="testMultiple" valor="3.0" vObtenido="1.5">
        <opcion id="a" esCorrecta="true">a) Enunciado Opcion1 Multiple</opcion>
      </pregunta>
      <pregunta id="desarrollo0" tipo="desarrollo" valor="4.0" vObtenido="false">Desarrollo1</pregunta>
      <pregunta id="recurso0" tipo="recurso" valor="3.0" vObtenido="false">url1</pregunta>
      <pregunta id="dr0" tipo="DR" valor="2.0" vObtenido="false" recurso="url11">Desarrollo11</pregunta>
    </seccion>
  </secciones>
</examen>
```

Ilustración 73: Respuestas de un alumno

Por cada alumno que realiza la actividad se guarda un fichero en formato xml en el directorio VideoDataDB/Respuestas/actividad siendo actividad el nombre indicado de la actividad. En ese directorio hay un fichero xml por cada alumno que ha realizado la actividad llamado idEmailAlumno.xml siendo el identificador del email del alumno y como se puede ver en la imagen anterior, se guardan las respuestas pero también los resultados obtenidos.

Ya que las preguntas tipo test son evaluadas de forma automáticas cuentan con un valor obtenido pero no se puede hacer lo mismo para las preguntas que son de desarrollo o recurso.

	A	B	C	D
1			jcasiano@al	pepe@alumno
2	desarrollo	4		
3	recurso0	3		
4	dr0	2		

Ilustración 74: Corrector de la actividad

Por este motivo, en el directorio VideoDataDB/Correcciones se crea un fichero llamado corrector+nombre siendo nombre el de la actividad. Es un fichero con formato xls que se utiliza para corregir las preguntas de tipo desarrollo, recurso y desarrollo-recurso. El Excel contiene en la primera columna los identificadores a las preguntas de este tipo, en la segunda columna la puntuación máxima a cada pregunta y en la primera línea contiene los alumnos que han realizado la actividad.

Corregir actividad

Para corregir la actividad hay que rellenar con las puntuaciones a cada pregunta y cada alumno del excel y proceder a realizar la evaluación.

	A	B	C	D
1			jcasiano@al	pepe@alumn
2	desarrollo	4	2	4
3	recurso0	3	2	3
4	dr0	2	2	1

Ilustración 75: Corrector rellenado

Se puede ver en la imagen de arriba como es una forma simple de evaluar a los usuarios las preguntas que no se pueden corregir de forma automática.

Corregir actividades automáticamente

```
kasiano@kasiano:~/Escritorio/proyectoVideoData/VideoData$ java -jar VideoDataAuto.jar -e testEvaluarActividades.txt
```

Ilustración 76: Actividades corregidas automáticamente

Para evaluar las actividades de forma automática hay que elegir la opción **-e** e indicar el nombre del fichero que contiene las actividades a evaluar. El fichero contiene en cada línea una o varias actividades a corregir, usando el siguiente formato.

-n nombreActividad [nombre2 nombre3]

Siendo nombreActividad el nombre de de la actividad a evaluar y pudiendo contener más actividades a evaluar, ya que lo que está entre corchetes es opcional. De esta forma la aplicación leerá el fichero y evaluará cada actividad que aparezca.

Para su correcto funcionamiento, las actividades han tenido que ser realizadas por algún alumnos ya que tiene que corregirse algún resultado de actividad. Y es conveniente que los excel correctores de cada actividad sean rellenados.

Se proporciona el fichero testEvaluaraActividades.txt como ejemplo.

Corregir actividad manualmente

```
---------------------------------------
Introduce opcion: 2
--- Corregir actividad ---
Nombre de la actividad (sin espacios): actividadConPreguntasVariadas
Calificado /home/kasiano/Escritorio/proyectoVideoData/VideoData/../VideoDataDB/Res
puestas/actividadConPreguntasVariadas/jcasiano.xml
Calificado /home/kasiano/Escritorio/proyectoVideoData/VideoData/../VideoDataDB/Res
puestas/actividadConPreguntasVariadas/pepe.xml
---------------------------------------
```

Ilustración 77: Actividad corregida

Para realizar la corrección se selecciona la tercera opción y una vez rellenado el corrector de la actividad (Ilustración 35) y guardado, ya podemos proceder a corregir la actividad a los alumnos. Es tan simple como introducir el nombre de la actividad.

jcasiano.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<examen>
  <metadatos>
    <datosPersonales nombreyapellidos="Jorge Casiano Bermejo" email="jcasiano@alumnos.unex.es" />
    <resultados puntuacionMax="13.0" puntuacionMin="6.5" puntuacionObtenida="7.0" notaTotal="53.846157" aprobadoTotal="true"
puntuacionDesRecMax="9.0" puntuacionDesRecObtenida="6.0" puntuacionTestMax="4.0" puntuacionTestObtenida="1.0" notaTest="25.0"
aprobadoTest="false" notaDesRec="66.66667" aprobadoDesRec="true" estaCorregido="true" />
  </metadatos>
  <secciones>
    <seccion nombre="Preguntas">
      <pregunta id="testSimple0" tipo="testSimple" valor="1.0" vObtenido="-0.5">
        <opcion id="a" esCorrecta="false">a) Enunciado Opcion1</opcion>
      </pregunta>
      <pregunta id="testMultiple0" tipo="testMultiple" valor="3.0" vObtenido="1.5">
        <opcion id="a" esCorrecta="true">a) Enunciado Opcion1 Multiple</opcion>
      </pregunta>
      <pregunta id="desarrollo0" tipo="desarrollo" valor="4.0" vObtenido="2.0">Desarrollo1</pregunta>
      <pregunta id="recurso0" tipo="recurso" valor="3.0" vObtenido="2.0">url1</pregunta>
      <pregunta id="dr0" tipo="DR" valor="2.0" vObtenido="2.0" recurso="url11">Desarrollo11</pregunta>
    </seccion>
  </secciones>
</examen>
```

Ilustración 78: Alumno corregido

Los ficheros xml que se encuentran en VideoDataDB/Respuestas/nombre de actividad han sido modificados insertando los valores a las preguntas de desarrollo y recurso y los resultados han sido también modificados.

Se puede ver la diferencia comparando la Ilustración 34 con la Ilustración 39 para ver como el vObtenido de las últimas preguntas a pasado de false al valor indicado en el corrector (Ilustración 36). Además la puntuación obtenida a pasado de un 1 a un 7.

Para evitar realizar la corrección varias veces sobre un mismo alumno, en cada alumno corregido se añade un atributo a su fichero xml estaCorregido=”true” para evitar que se corrija más de una vez. Si se deseara volver a corregir una alumno, habrá que modificar manualmente el fichero xml eliminando ese atributo y modificando las notas para restaurarlas.

Almacenamiento de los resultados en vídeo

El último paso es guardar los resultados de cada alumno en un vídeo. Para ello se utilizará otro módulo diferente.

```
kasiano@kasiano:~$ cd Escritorio/proyectoVideoData/VideoData
kasiano@kasiano:~/Escritorio/proyectoVideoData/VideoData$ cd Python/
kasiano@kasiano:~/Escritorio/proyectoVideoData/VideoData/Python$ python main.py
Manejo de datos (VideoData)
----------------------------
Opciones
1.- Crear video
2.- Obtener todos los ficheros
3.- Obtener un fichero
4.- Obtener examen
0.- Salir
----------------------------
Escoge una:
```

Ilustración 79: Consola para gestionar los datos

En la imagen anterior se puede ver como se accede a este módulo que está separado del módulo de creación.

```
----------------------------
Escoge una: 1
Introduce el nombre del modelo: actividadConPreguntasVariadas
Indexando 2 ficheros
Video con 1 imagenes con informacion
Creating video ../../VideoDataDB/VideosDB/actividadConPreguntasVariadas.avi
----------------------------
```

Ilustración 80: Crear vídeo de almacenamiento

De esta forma, seleccionando la primera opción e introduciendo el nombre de la actividad se procede a guardar los datos de los alumnos. Concretamente se guardan los ficheros xml que se encuentran en VideoDataDB/Respuestas/nombre siendo nombre actividadConPreguntasVariadas.

Esto creará un vídeo que es usado como base de datos en el directorio VideoDataDB/VideosDB con el nombre de la actividad y la extensión AVI. De esta forma ya se puede borrar el directorio VideoDataDB/Respuestas/nombre de actividad ya que los resultados han sido guardados en el vídeo.

Recuperación de los resultados del vídeo

Para recuperar los resultados a partir del vídeo se utiliza la misma consola y se ofrecen dos opciones, recuperar un alumno o recuperar todos. Primero recuperaremos al alumno cuyo identificador de email es jcasiano.

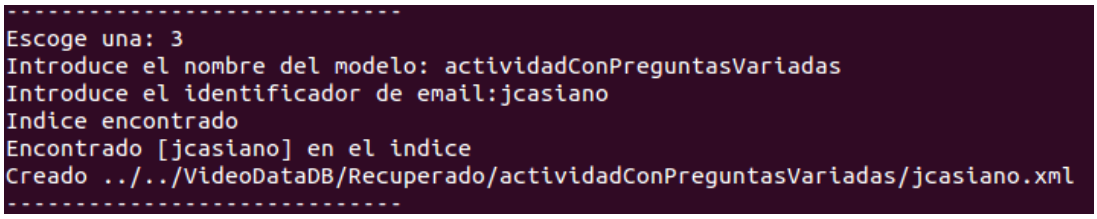


Ilustración 81: Recuperación de un alumno

Para recuperar un alumno se elige la tercera opción, se elige el nombre de la actividad y posteriormente el identificador el alumno a recuperar, en este caso jcasiano. Esta acción extrae del vídeo los resultados de dicho alumno y lo guarda en el directorio VideoDataDB/Recuperado/actividadConPreguntasVariadas que es creado e insertado el documento jcasiano.xml.

Si por el contrario se quiere hacer una carga masiva y extraer los resultados de todos los alumnos que han realizado la actividad.

Ilustración 82: Recuperación de todos los alumnos

Como se puede ver en la imagen anterior, se selecciona la segunda opción y se indica el nombre de la actividad. De esta forma en el directorio VideoDataDB/Recuperado/actividadConPreguntasVariadas se guardan los dos ficheros que corresponden con los dos alumnos que han realizado la actividad.

Con la opción 4 se recuperará únicamente el contenido del examen o actividad, es decir todas las preguntas con sus opciones, soluciones correctas y los valores de cada pregunta.

Capítulo 6.- Reflexiones acerca de este proyecto

Sobre el proyecto

El nacimiento de este proyecto

Problemas durante el desarrollo

Al tratarse de un proyecto donde se utilizan distintas tecnologías y que puede ser considerado complejo se han tenido algunos problemas durante el desarrollo unos más graves y otros menos que finalmente se han podido resolver. En esta sección se comentarán los problemas durante el desarrollo en cada parte:

Problemas en la creación de contenido

Los primeros problemas surgidos han sido en relación con el formato de los ficheros de entrada que son ficheros Excel con formato xls. Había problemas con la lectura y creación de estos ficheros que finalmente se ha solucionado utilizando la librería **jxl.jar** que permite manejar estos ficheros.

Otro problema surgido era como crear los modelos y los ficheros xml que representan los hipervídeos de una forma óptima, la solución ha consistido en el uso de **JDOM** para crear estos ficheros. Se ha elegido esta tecnología que es similar a DOM pero está creada para usarse de forma nativa en java aprovechando mejor los recursos.

Tras crear las actividades como modelos, el siguiente paso es aplicar el template Acceleo de una forma automática. Esta tarea resultó compleja debido a que Acceleo tiene su propio entorno de ejecución. La solución consistió en añadir al *Java Build Path* el proyecto **VideoDataAcceleo** y la librería **Acceleo Runtime** que permite la ejecución desde java de templates Acceleo. Después se utilizó la clase URI para crear la URI hasta el metamodelo desarrollado y llamar al template.

Problemas en la creación de VideoDatas

Denominando VideoData a vídeos de almacenamiento de información que son usadas como bases de datos, existieron varios problemas que poco a poco fueron resueltos.

El **primer problema era utilizar una tecnología que permitiese acceder a un frame del vídeo de forma directa** sin tener que acceder a los frames anteriores. La

solución fue **el uso de OpenCV** que **permitía esta tarea que resulta fundamental para obtener los ficheros de una forma eficiente**.

Una vez elegida la tecnología a usar surgió el problema más grave que consistía en que **los codecs que soporta OpenCV aunque en la documentación aparezcan como sin pérdida o *lossless* realmente no era así**. La imagen guardada y la recuperada del vídeo no era exactamente igual por ello tuve que buscar una solución a este importante problema porque al tratar con información textual el contenido guardado debe ser exactamente igual al introducido.

La primera tarea fue ir **probando cada codec y analizando cual era la correspondencia entre una imagen antes de guardarla y la misma después de obtenerla del vídeo**. Y si esa correspondencia era constante, es decir si siempre se obtenía la misma información para el mismo contenido guardado. El resultado fue que **se encontró el codec HYFU donde la correspondencia** o diferencia entre la imagen original y la obtenida del vídeo **era constante, es decir para un valor concreto siempre se obtenía el mismo valor**.

Una vez encontrado el codec idóneo se buscó la manera de poder obtener finalmente la información del vídeo de forma lo más eficiente posible y en el menor espacio posible.

La solución fue utilizar por cada frame de información un frame de corrector de ruido para reconstruir el frame original guardado y poder recuperar la información exacta. Esta solución es eficiente aunque cuenta con el problema del espacio ya que **el vídeo utiliza el doble de espacio para almacenar la información, esto podrá ser resuelto si se desarrolla algún codec compatible con OpenCV que realmente sea lossless**.

Con la solución actual el coste de almacenar un byte de información son dos bytes y medio. De estos dos bytes y medio, **dos bytes son para reconstruir el byte correcto y el medio byte es el coste de compresión** y tratamiento de la imagen y el vídeo **debido a que no ocupa el mismo espacio el texto plano que una imagen con la misma información**.

Planificación del proyecto

Trabajo futuro

Conclusiones

Bibliografía

[1] «Welcome to Flubaroo». [En línea]. Disponible en: http://www.flubaroo.com/. [Accedido: 16-nov-2015].

[2] «MOOC - educaLAB». [En línea]. Disponible en: http://educalab.es/mooc. [Accedido: 10-nov-2015].

[3] Marco Cuenca, G., Arquero Avilés, R., Ramos Simón, LF., Cobo Serrano, S. «Análisis de características de los Cursos en Línea Masivos y Abiertos (MOOCs): propuesta de aplicación en escenarios de aprendizaje en el área de Documentación.» Disponible en: http://www.congresosweb.info/index.php?option=com_mtree&task=att_download&link_id=136&cf_id=24. [Accedido: 10-nov-2015]

[4] «DONALD CLARK: Taxonomy of 8 Types of MOOC», Ticeduforum. Disponible en: http://ticeduforum.akendewa.net/donald-clark-taxonomy-of-8-types-of-mooc/. [Accedido: 10-nov-2015]

[5] Cambero Almenara, J., Llorente Cejudo, MC. y Vázquez Martínez, AI. «Las tipologías de MOOC: su diseño e implicaciones educativas», Universidad de Sevilla.Disponible en: http://digibug.ugr.es/bitstream/10481/31663/1/rev181ART1.pdf. [Accedido: 7-nov-2015]

[6] Mayra Eugenia Armijos, «DEMOCRATIZACIÓN DE LA EDUCACIÓN A DISTANCIA MEDIADA A TRAVÉS DE MOOCS». Disponible en: http://www.researchgate.net/profile/Mayra_Armijos/publication/281034212_DEMOCRATIZACIN_DE_LA_EDUCACIN_A_DISTANCIA_MEDIADA_A_TRAVS_DE_MOOCS/links/55d2143c08ae7fb244f40f21.pdf [Accedido: 8-nov-2015]

[7] «Los cursos online MOOC, una tendencia formativa en auge - educaweb.com». [En línea]. Disponible en: http://www.educaweb.com/noticia/2013/02/06/cursos-online-mooc-tendencia-formativa-auge-5978/. [Accedido: 8-nov-2015].

[8] Oliver, M., Hernández-Leo, D., Daza, V., Martín, C. y Albó, L. «Panorama actual de los Cursos Masivos Abiertos en Línea en las universidades españolas».Disponible en: http://www.catedratelefonica.upf.edu/wp-content/uploads/2014/02/MOOCs-en-Espa%C3%B1a1.pdf [Accedido: 7-nov-2015]

[9] «El Gobierno catalán otorga ayudas para tres MOOC de la UPC — Sala de Premsa - Universitat Politècnica de Catalunya (UPC)». [En línea]. Disponible en: http://www.upc.edu/saladepremsa/al-dia/mes-noticies/la-generalitat-otorga-ayudas-para-tres-mooc-de-la-upc. [Accedido: 10-nov-2015].

[10] A. Pedreño Muñoz, L. Moreno Izquierdo, A. Ramón Rodríguez, y P. Pernías Peco, «La crisis del modelo actual. Los MOOC y la búsqueda de un modelo de negocio», The crisis of the current model. MOOCs and search for a business model, 2013.

[11] Roig Vila, R., Mengual – Andrés, S. y Suárez Guerrero, C. «EVALUACIÓN DE LA CALIDAD PEDAGÓGICA DE LOS MOOC», Universidad de Alicante. Universidad de Valencia. [En línea]. Disponible en: http://www.ugr.es/~recfpro/rev181ART2.pdf [Accedido: 11-nov-2015].

[12] «El 90% de los estudiantes no terminan sus cursos online», Noticias Universia España. [En línea]. Disponible en: http://noticias.universia.es/en-portada/noticia/2014/01/16/1075157/90-estudiantes-no-terminan-cursos-online.html. [Accedido: 17-nov-2015].

[13] E. E. País, «La deserción puede con los cursos 'online'», EL PAÍS, 12-ene-2014. [En línea]. Disponible en: http://economia.elpais.com/economia/2014/01/10/actualidad/1389360489_728192.html. [Accedido: 17-nov-2015].

[14] P. por E. de UniMOOC, «Coursera apuesta por las certificaciones de pago | Observatorio MOOC». [En línea]. Disponible en: http://blogmooc.iei.ua.es/2013/02/coursera-apuesta-por-las.html. [Accedido: 17-nov-2015].

[15] «Harvard se aleja de los MOOC y lanza una plataforma de cursos virtuales pagos», iProfesional. [En línea]. Disponible en: http://www.iprofesional.com/notas/183602-Harvard-se-aleja-de-los-MOOC-y-lanza-una-plataforma-de-cursos-virtuales-pagos. [Accedido: 17-nov-2015].

[16] E. E. País, «¿Qué fue de la revolución MOOC?», EL PAÍS, 09-oct-2014. [En línea]. Disponible en: http://economia.elpais.com/economia/2014/10/08/actualidad/1412783861_083138.html. [Accedido: 15-nov-2015].

[17] García-Valcárcel, A. (2008). «El hipervídeo y su potencialidad pedagógica». Revista Latinoamericana de Tecnología Educativa. [En línea]. Disponible en: http://relatec.unex.es/article/view/415/347.[Accedido: 15-nov-2015].

Anexo 1: Instalación del sistema

Instalación para usuarios

El sistema VideoData se basa en tres pilares que usan distintas tecnologías, pero se pueden usar de forma independiente cada una de ellas. Lógicamente el potencial radica en usar la tres de forma fusionada pero el diseño está realizado para permitir el uso de cada módulo de forma independiente. Por ello **si se pretende usar el sistema completo es obligatorio usar un sistema operativo Unix** ya que es el único sistema que soporta todas las funcionalidades.

La instalación de la aplicación consiste en descomprimir el proyecto y pegarlo en un directorio siendo recomendable que la ruta no tenga espacios. Y posteriormente basta con configurar el fichero configuracionRuta.txt con la ruta absoluta hasta el proyecto *proyectoVideoData/VideoData* en la primera línea y pegar este fichero en los siguientes directorios:

- ***proyectoVideoData/VideoData***
- ***proyectoVideoData/VideoDataWeb***
- Si se usa **servidor sin eclipse**: *tomcat/bin*
- Si se usa **servidor con eclipse**: *eclipse/*

Además de instalar correctamente la aplicación es necesario tener algunas tecnologías instaladas para cada módulo del sistema.

Creación de contenido

La creación del contenido pueden ser realizado desde sistemas Unix, Windows y Mac OS X, es necesario tener instalado java y disponer de un navegador que permita el uso de HTML5.

Puede comprobar si tiene instalado java usando el comando *java -versión* en línea de comando, siendo conveniente tener la versión 7 u 8.

>java -versión

Si no está instalado java, se puede descargar e instalar fácilmente aquí.

En cuanto los navegadores, cualquiera que este actualizado funcionará. Únicamente no es soportado html5 por versiones antiguas de Internet Explorer.

Prueba de la instalación de creación de contenido

Prueba de la generación de actividades

Si se quiere probar que este módulo está correctamente instalado, en el directorio *proyectoVideoData/VideoDataDB/test/entradaDatos/actividades* hay un conjunto de ficheros de entrada que permiten crear exámenes con preguntas de distintos tipos y otros con preguntas variadas. Estos ficheros se pueden copiar en el directorio de entrada de datos de exámenes que es *proyectoVideoData/EntradaDatos/Actividades* y ejecutar el comando.

> ***java -jar VideoDataAuto.jar -a GenerarActividadesTest.txt***

Este comando generará las actividades de tipo test de los ficheros copiados, obteniendo el examen en la carpeta *proyectoVideoData/VideoData/model* pudiendo compararse estos ficheros con los ficheros que se encuentran en la ruta *proyectoVideoData/VideoDataDB/test/modelos.*

También se pueden comparar los ficheros que representan la actividad en la aplicación web, estos se encuentran en *proyectoVideoData/VideoDataWeb/WEB-INF/* y se pueden comparar con los ficheros con el mismo nombre de la ruta *proyectoVideoData/VideoDataDB/test/actividades*. Y la lógica de la actividad generada se encuentra en el *proyectoVideoData/VideoDataWeb/src/videodata/actividad* en los directorios */enunciados* y */servlet* pudiendo ser comparados con los ficheros que tienen el mismo nombre en el directorio *proyectoVideoData/VideoDataDB/test/actividades.*

Prueba de la creación de hipervídeos

Para probar la creación de los hipervídeos se pueden copiar los ficheros que se encuentran en la carpeta *proyectoVideoData/VideoDataDB/test/entradaDatos/popcorn* y pegarlos en *proyectoVideoData/EntradaDatos/Popcornjs*. Después ejecutar el comando.

> ***java -jar VideoDataAuto.jar -h crearHipervideoTest.txt***

Que creará dos hipervídeos, estos se encuentran en el directorio *proyectoVideoData/VideoDataWeb/WebContent/video* llamados *hipervideoDePruebaHTTP.html* y *hipervideoDePruebaInterno.html*, estos ficheros pueden ser comparados con los encontrados en *proyectoVideoData/VideoDataDB/test/hipervideo.*

Evaluación

Para la evaluación será necesario el uso de un servidor de aplicaciones web como Tomcat. Si no disponemos de una servidor dedicado, dependiendo de los conocimientos técnicos puede resultar conveniente el uso de un servidor local y usar este para realizar la evaluación y permitir el uso del sistema por parte de los alumnos.

Para un uso sencillo lo más conveniente es usar Tomcat desde el IDE Eclipse e importar la aplicación web donde se generan las actividades.

Primero hay que descargar Eclipse J2EE para poder manejar la aplicación web mejor y Apache Tomcat 7 para desplegar la aplicación web.

Después descomprimimos lo descargado, abrimos eclipse y procedemos a importar la aplicación web que se encarga de gestionar las actividades creadas. Esto se realiza pulsando con el *botón derecho* sobre la parte izquierda (*Project Explorer*) → *Import* → *Import …* → *General* → *Existing Projects into Workspace* → *Browse (Select root directory)* y seleccionamos el proyecto **VideoDataWeb** y pulsamos *Finish*.

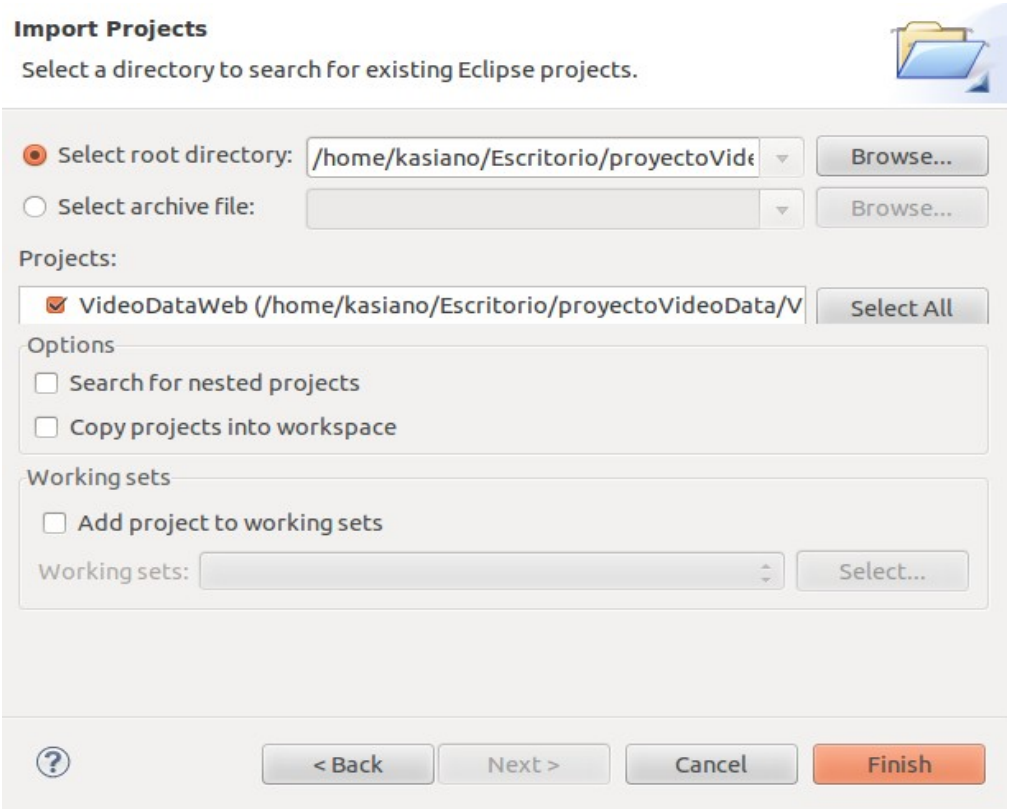


Ilustración 83: Importar VideoDataWeb

Tenemos dos opciones para desplegar el servidor, sin eclipse o con eclipse. La elección dependerá de la comodidad que sienta el profesor con ambos sistemas para ello es conveniente ver como se despliega el servidor y se ejecuta la aplicación web de las dos formas.

Aunque es cuestión de gustos personales, **se recomienda que si en el sistema se va a estar usando continuamente la creación de contenido resulta más conveniente desplegar el servidor usando eclipse** por comodidad. **Si** por el contrario **los contenidos son creados pocas veces**, normalmente una vez al principio, **se recomienda la opción de desplegar el servidor sin eclipse**.

Desplegar en servidor (sin eclipse)

Si no queremos que el servidor dependa de eclipse, ya que disponemos de un servidor para desplegar la aplicación más profesional, hay que crear el fichero WAR para ello hay que pulsar con *botón derecho sobre VideoDataWeb → Export → WAR file* → Damos un nombre y proporcionamos el directorio donde se creará.

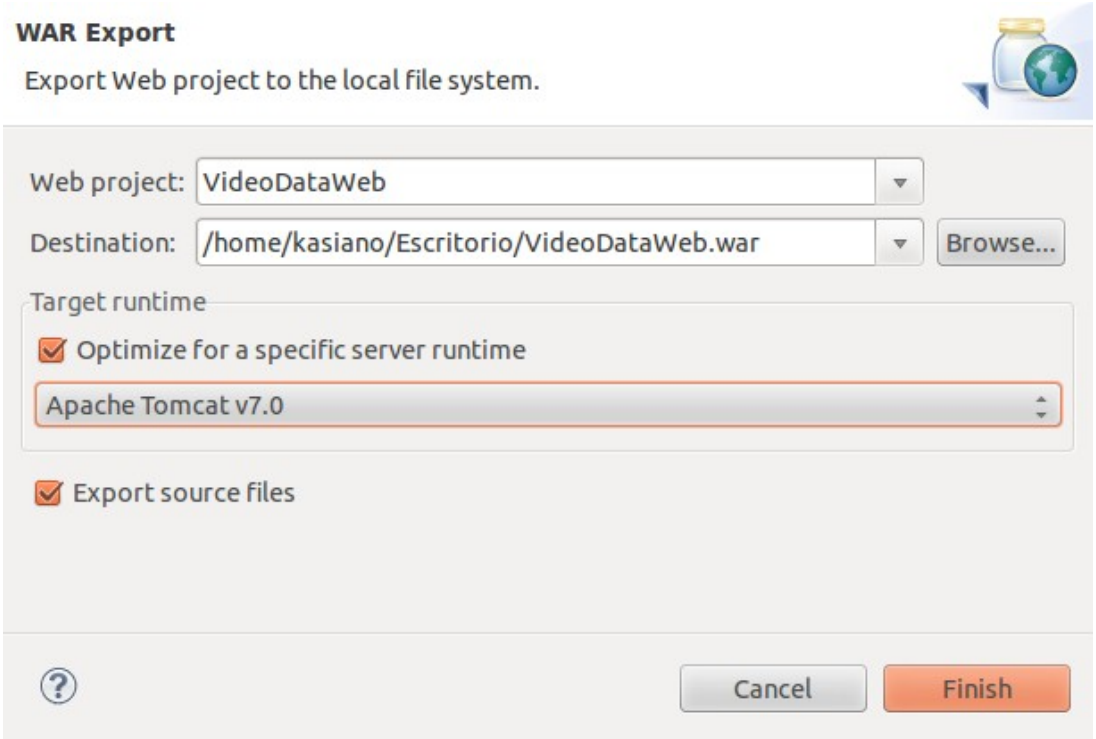


Ilustración 84: Exportar WAR

Una vez creado el fichero WAR, tenemos que pegarlo en el directorio *Tomcat/webapps* siendo *Tomcat* la carpeta descomprimida obtenida a partir de descomprimir el fichero descargado.

Desplegar en servidor (con eclipse)

El siguiente paso es configurar eclipse para usar Tomcat, esto se realiza pulsando sobre la pestaña *Window → Preferences → Server → Runtime Environment* y pulsamos en *Add* seleccionando Apache Tomcat v7 y *next*. Después sobre *Browse...* y elegimos la carpeta que se ha creado a partir de descomprimir Tomcat.

Después nos vamos a la pestaña de abajo llamada Servers y pulsamos para crear uno nuevo, *seleccionamos Tomcat v7 → Finish*. Pulsamos sobre el server que acabamos de crear y tenemos configurar la parte de *Server Locations* eligiendo la segunda opción (Use Tomcat installation) y en *Deploy path* modificamos "wtpwebapps" por **webapps** y guardamos.

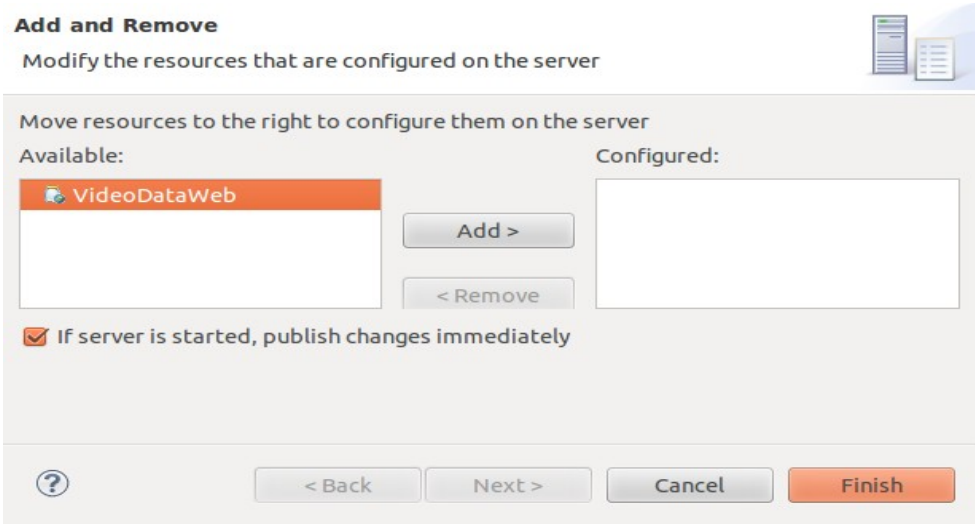


Ilustración 85: Desplegar VideoDataWeb

Para desplegar la aplicación pulsamos sobre el server con *botón derecho → Add and Remove...* seleccionamos la aplicación, pulsamos sobre *Add>* y *Finish* como aparece en la imagen anterior. De esta forma ya tenemos desplegados la aplicación web, pulsando sobre el server con *botón derecho → Start* o directamente el play verde.

Ilustración 86: Despliegue realizado con éxito

Como se puede observar en la imagen anterior ya tenemos la aplicación desplegada y podemos proceder a trabajar con las actividades, en este caso es una actividad de prueba a la que podemos acceder a través del navegador.

Prueba de la instalación de la evaluación

Para probar la evaluación hay que crear primero la actividad preguntasTipoVariadoReducido, si se ha realizado la **Probando la instalación de creación de contenido** ya estará creado, aunque por defecto ya viene el examen preguntasTipoVariadoReducido por defecto. Si de todas formas se quiere crear de nuevo hay que ejecutar el siguiente comando.

> ***java -jar VideoDataAuto.jar -a GenerarActividadesTest.txt***

De esta forma se generará la actividad, es siguiente paso es desplegar el servidor con la aplicación **VideoDataWeb.**

Una vez desplegado se puede probar la evaluación realizando el examen, se encontrará en la url *http://localhost:8080/VideoDataWeb/preguntasTipoVariado Reducido.html* y realizar el examen.

Si se quiere comparar con un fichero, hay que introducir o seleccionar dependiendo el tipo de pregunta en todos los campos una 'a' e indicar el email *a@a*. Al enviar el examen se creará un fichero en el directorio *proyectoVideoData/VideoDataDB/ Respuestas/preguntasTipoVariadoReducido* llamado *a.xml* que contiene las respuestas y resultados del examen recientemente realizado. Este ficheros se puede

comparar con *proyectoVideoData/VideoDataDB/test/respuestas/preguntasTipoVariadoReducido/a.xml* y debe ser igual. Además se han tenido que crear el fichero que se encarga de las correcciones manuales *proyectoVideoData/VideoDataDB/Correcciones/correctorpreguntasTipoVariadoReducido.xls* con el usuario introducido junto a las preguntas que deben ser corregidas y el fichero que almacena a todos los usuarios con sus respuestas *proyectoVideoData/VideoDataDB/Respuestas/preguntasTipoVariadoReducido.xls* que contiene las respuestas de todos los usuarios (en este caso las respuestas de a@a).

El siguiente paso es dar unas puntuaciones a las preguntas usando el corrector *correctorpreguntasTipoVariadoReducido.xls*, después guardar el fichero y ejecutar el comando.

> ***java -jar VideoDataAuto.jar -e CorregirExamen.txt***

Modificando con las puntuaciones indicadas en el excel corrector el fichero *a.xml*.

Almacenamiento en vídeo (VideoData)

Para poder almacenar los resultados de los alumnos en vídeo y ser usados como base de datos, **es obligatorio usar un sistema operativo Unix** como puede ser Ubuntu.

Hay que tener instalado Python 2.7, puedes comprobar si está instalado escribiendo en una consola *python --versión* si no es la versión 2.7 habrá que instalarlo para ello seguir las siguientes instrucciones:

Instalar algunas dependencias:

```
sudo apt-get install build-essential checkinstall
sudo apt-get install libreadline-gplv2-dev libncursesw5-dev
libssl-dev libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev
```

Descargar python:

```
cd ~/Downloads/
wget http://python.org/ftp/python/2.7.5/Python-2.7.5.tgz
```

Descomprimir el fichero descargado.

Ir al directorio:

```
cd Python-2.7.5
```

Instalar:

```
./configure
make
sudo checkinstall
```

De esta forma ya estaría instalado Python 2.7, para comprobarlo repetimos el proceso escribiendo *python* --versión.
También es necesario instalar OpenCV, esto se realizará desde línea de comando y se pueden seguir las siguientes instrucciones.

Prueba de instalación del almacenamiento VideoData

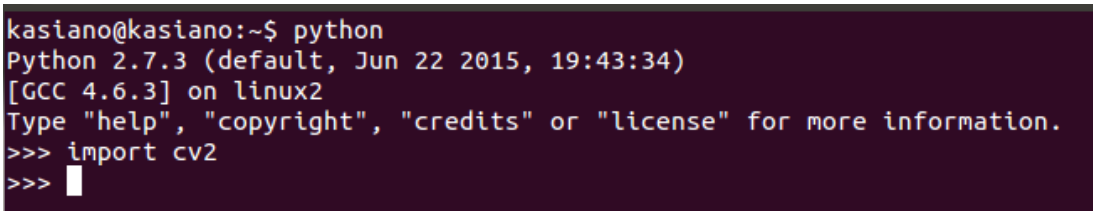


Ilustración 87: Comprobación de la instalación de OpenCV

Una vez realizada la instalación de OpenCV podemos comprobar que está correctamente instalado como aparece en la imagen anterior obteniendo el mismo resultados sin fallos al realizar el import.

También se puede probar si se ha realizado algún examen siguiendo los pasos **Guardando los resultados en vídeo** y **Recuperando los resultados del vídeo**.

Instalación para desarrolladores

Al usarse tantas tecnologías distintas es necesario la instalación de distintos componentes software. La parte de creación de contenido, evaluación y la aplicación web ha sido desarrollada usando java y el módulo VideoData usando Python combinado con OpenCV.

Antes de proceder con la instalación que necesita el desarrollador, **es imprescindible tener todas las tecnologías instaladas indicadas en la instalación para usuarios.**

Para el desarrollo del sistema íntegro, es imprescindible usar un sistema operativo Unix, en el caso del desarrollador se ha usado Ubuntu 12.04.

Creación de contenido, evaluación y aplicación web

Partiendo de que ya se tiene instalado java, con el objetivo de facilitar el entorno necesario se proporcionará en el CD del TFG en entorno Eclipse para el desarrollo de la creación de contenido, la evaluación y la aplicación web.
Por si no se dispone del CD, se enumerará los pasos a seguir para tener el entorno Eclipse con todos sus componentes:

Descargar Eclipse Modeling Tool (Version Juno), y descomprimir. Este eclipse es necesario para realizar el metamodelado de las actividades. Después de descomprimir y abrir eclipse.

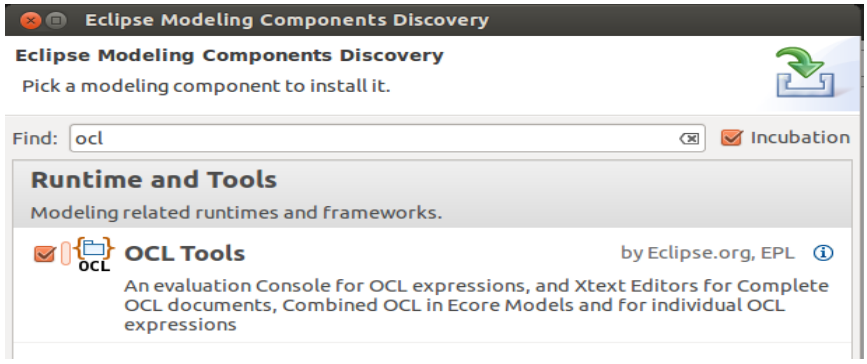


Ilustración 88: Instalación de OCL

Hay que instalar **OCL** para definir las restricciones sobre el metamodelo, se puede hacer de forma sencilla pulsando en la pestaña *Help → Installing Modeling*

Components y buscar las tecnologías indicadas e instalarla.

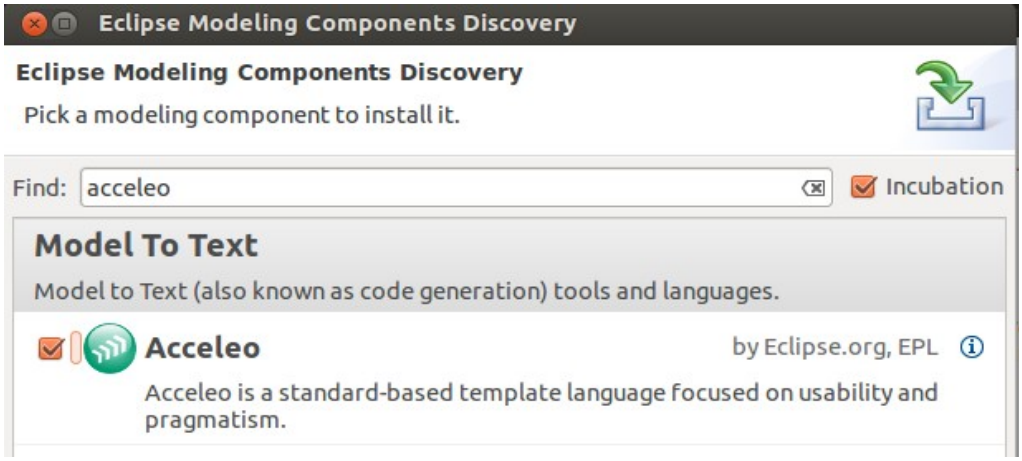


Ilustración 89: Instalación de Acceleo

También es necesario instalar **Acceleo** para crear las transformaciones de modelo a texto *(m2t)*. El proceso es el mismo que el anterior. De esta forma ya estaría la parte de modelado instalada.

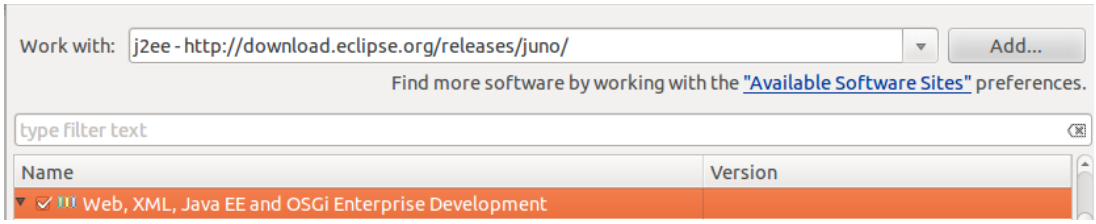


Ilustración 90: Instalación desarrollo J2EE en eclipse

Solo quedaría instalar los componentes para realizar el desarrollo de la aplicación web, pulsando sobre *Help → Install new software …* e introducir lo que aparece en la imagen anterior y seleccionar la parte marcada. Después de realizar la instalación ya está este entorno preparado.

VideoData

La instalación de esta parte está indicada en el manual de usuario detalladamente. Para poder almacenar los resultados de los alumnos en vídeo y ser usados como base de datos, **es obligatorio usar un sistema operativo Unix** como puede ser Ubuntu.

Hay que tener instalado Python 2.7, puedes comprobar si está instalado escribiendo en una consola *python --versión* si no es la versión 2.7 habrá que instalarlo y también hay que tener instalado OpenCV.

Para desarrollar esta parte en Python ha sido usado Sublime Text 3 aunque puede usarse cualquier editor de texto.

Lista de figuras