# Sistemas informáticos
# Curso 2004-2005

## Entorno de clase virtual en Internet

Sara Cantos Galán
Roberto Álvarez Sánchez
José-Luis Parrilla Bejerano

Dirigido por:
Baltasar Fernández-Manjón
Dpto. Sistemas Informáticos y Programación

---

Facultad de Informática
Universidad Complutense de Madrid

# Sección I. Introducción y motivación

# 1.- Acerca de este documento.

El proyecto descrito en el presente documento consiste en la implementación de un módulo que añade una serie de funcionalidades al proyecto de "e-learning", previamente existente, denominado <e-Aula>. Además de esta parte de implementación, el trabajo consta de un importante apartado documental donde se describen con detalle dos de los estándares más influyentes dentro del campo del aprendizaje virtual: ADL-Scorm y LIP.

Los dos objetivos más prioritarios de este proyecto son, por un lado, dar un paso más en la estandarización de los sistemas "e-learning", aspecto que consideramos de vital importancia para la potenciación de los mismos. Para ello hemos realizado un trabajo de documentación acerca de los dos estándares anteriormente comentados y hemos hecho partícipe al segundo de ellos (LIP) en nuestra implementación.

Por otro lado, incorporar la adaptación de los contenidos del curso, es decir, en la forma en que éstos se presentan al alumno dependiendo de sus preferencias y del dispositivo mediante el cual se conectan al sistema. Entendemos que los alumnos son los usuarios que justifican la realización de los sistemas de "e-learning", por ello, consideramos que este tipo de sistemas no deben sólo basarse en la presentación de contenidos de alta calidad educativa, sino también proporcionar al usuario la posibilidad de poder editar la forma en que estos contenidos son presentados en función de sus preferencias, pero siempre limitados por las características del dispositivo con el cual accedan a dichos contenidos.

Además de estos objetivos prioritarios, nuestro proyecto incorpora también mecanismos de seguimiento para facilitar a los tutores su labor de evaluación sobre los alumnos, como son la generación de estadísticas relativas principalmente a la interactuación de los alumnos con el sistema así como un módulo para simplificar la comunicación vía e-mail entre el tutor y sus alumnos.

El propósito de este proyecto es que sus contenidos sean incorporados, si no en su totalidad, sí al menos en su gran mayoría, al sistema <e-Aula> por razones de *"necesidad"*. Es decir, que el sistema <e-Aula> siga los cauces de estandarización y accesibilidad fijados por sus desarrolladores originales y que éstos se vean reflejados en nuestro proyecto. Esto supondría que hemos alcanzado las metas que al comienzo de este curso académico nos presentó nuestro director de proyecto, el profesor Baltasar Fernández-Manjón y con ello, que este trabajo de Sistemas Informáticos ha supuesto algo más que una labor estrictamente académica.

## 2.- About this document.

The Project described in this document consists of the implementation of a module which adds several functionalities to the e-learning project, that previously existed, called e-aula. Besides the implementation part, this work has an important documentary section where are described the most influent standards in virtual learning area: ADL-Scorm and LIP.

There are two principal objectives in this project: first of all to improve the standardization of e-learning systems, something we consider very significant in order to boost them. As for this, its includes a documentary work on these two standards and also the use of LIP in our implementation.

Secondly, we also wanted to adapt the course contents to the preferences of the students when they use them, since we do not consider the high quality contents as the only one aim of e-learning systems, but also to provide the users different options to edit the way these contents appear (obviously this possibility is limited by the device characteristics).

Finally, our project tries to make the students evaluations easier, providing stadistics related to the interplay between students and system and making simplier the e-mail communication between tutor and students.

The main purpose of this project is to incorporate as many contents as we can to the e-aula project, paying special attention to the standardization and accessibility set by the original developers. In this way we would have achieved the aims which our director, professor Baltasar Fernández Manjón, suggested, and at the same time, this work would have been something more than a simple academic task.

## 3.- E-learning.

*"La enseñanza asistida por computadora es la organización y combinación de los recursos educativos y tecnológicos para permitir el teleaprendizaje por parte de los alumnos"* (Lanfranco, 1993)

La enseñanza virtual, como consecuencia de las nuevas tecnologías (Internet es actualmente el mayor medio de transmisión de información), está experimentando un auge creciente en los últimos años.

Un gran número de empresas, instituciones y particulares han visto en ella no sólo un sistema eficaz desde un punto de vista didáctico, que conlleva los mismos pasos que los de una formación presencial: detección de necesidades, diseño de contenidos, plan de actuación, métodos de ejecución y valoración de resultados, sino también un nuevo medio para poder superar algunas de las deficiencias educativas de tiempos pasados:

- Flexibilidad absoluta de horarios. Los alumnos pueden realizar su aprendizaje a cualquier hora y durante el tiempo que deseen.

- Procuran una comunicación totalmente bidireccional tanto entre alumnos como entre profesor y alumnos, alejándonos así del tópico de "enseñanza aislada o en solitario". Herramientas como el correo electrónico, foros, videoconferencias...fomentan la participación del alumno en el proceso formativo. Incluso Hiltz (Hiltz, 1994) afirmó que dentro de un entorno virtual, la participación de los alumnos puede llegar a ser incluso mayor que en un aula convencional.

- La universalidad de estos sistemas hacen que el alumno pueda escoger entre materiales preparados por los mejores especialistas dentro de una gran diversidad de temáticas.

Paralelamente comienzan a surgir una serie de inconvenientes como el aumento en la hetereogeneidad de productos e-learning o el espectacular aumento de la información que dificulta en gran medida las relaciones entre los distintos sistemas. La situación comienza a demandar una unificación de criterios a la hora de crear contenidos, implantar plataformas y almacenar datos.

## 4.- Estandarización

*"La estandarización de las tecnologías aplicadas al aprendizaje pretende posibilitar la reutilización de recursos educativos y la interoperabilidad entre sistemas software heterogéneos"* (Anido, 2002)

La estandarización aplicada a los sistemas e-learning surge como una necesidad para aplacar los problemas de transferencia, generalmente de contenidos, entre este tipo de sistemas. Estos estándares proporcionan una gran flexibilidad a las soluciones e-learning, tanto en contenido como en infraestructura.

Su irrupción en el mundo de la educación a distancia ha supuesto la utilización de una metodología más coherente a la hora de empaquetar recursos y contenidos, tanto para los estudiantes como para los desarrolladores; solucionándose así las carencias de las anteriores metodologías, incapaces de ofrecer garantías en los objetivos de accesibilidad, interoperabilidad, durabilidad y reutilización de los materiales pedagógicos basados en Web.

La estandarización de estos sistemas ha supuesto una confluencia de tecnologías muy importante para los usuarios de estos sistemas puesto que un producto construido sobre los cimientos de la estandarización no quedará obsoleto a corto plazo, lo cual proporcionará una garantía para sus inversores y para el futuro del e-learning.

# Sección II. El proyecto <e-Aula>

# 1.- Descripción general del sistema.

<e-Aula> es un proyecto desarrollado por el departamento de Sistemas Informáticos y Programación de la Universidad Complutense de Madrid.

El objetivo principal del proyecto es crear un entorno de aula virtual que, mediante los lenguajes de marcado y las tecnologías Web asociadas, especifique e implemente un sistema de e-learning de acuerdo al modelo de referencia de los estándares educativos. La metodología de creación del aula virtual pretende que el aula sea flexible, abierta y ampliable, de modo que se mejore la gestión, el acceso, la interactividad y la utilidad de la información educativa proporcionada mediante la red.

<e-Aula> nació con el objetivo de analizar el potencial de los estándares educativos en la construcción de entornos de aprendizaje personalizado. Más concretamente en su comienzo sirvió cómo evaluador de las especificaciones IMS y EML mediante la implementación de funciones básicas de cada uno de ellos. IMS ha englobado a EML, por lo que <e-Aula> se centró únicamente en IMS poniendo así a prueba la propuesta en un sistema real de enseñanza con aplicación en la universidad.

El sistema incluye gestiones administrativas, el proceso de estudio de los alumnos, la elaboración de los contenidos por parte de los tutores, la evaluación de los conocimientos adquiridos y la comunicación entre tutores y alumnos.

El sistema actual se basa en el estándar ADL-SCORM, ampliado y contrastado con las propuestas de IMS. No obstante el objetivo del proyecto es cumplir con las normas del estándar desarrollado por ISO, una vez que haya sido completado.

# Sección III. Fundamentos del proyecto

# 1.- Fundamentos técnicos.

## 1.1.- El estándar ADL-SCORM

### 1.1.1.- Primera visión de SCORM 2004

El departamento de defensa Americano lanza en 1997 la iniciativa ADL (Avanced Distributed Learning), con la intención de desarrollar la educación a distancia vía Web, con gran calidad y que se adaptara a las necesidades individuales de cada alumno.

En el momento en que la Web aparece en los gobiernos, universidades, industria y demás sectores de la sociedad es cuando se entiende la idea de SCORM. Debido al gran potencial de Internet era necesario crear un estándar bajo el cual todo el mundo pudiera trabajar en el mismo camino. Basado en documentos y estudios anteriores SCORM quiso poner la Web como medio principal del desarrollo de la enseñanza personalizada.

SCORM proporciona un API de comunicación para la interacción de los estudiantes con los “objetos” que componen los cursos, define un modelo de datos para representar la información, un conjunto de meta-datos utilizados para la descripción de los contenidos educativos y un grupo de reglas de secuencia para guiar el desarrollo de las sesiones.

SCORM continua desarrollando los fundamentos técnicos de e-learning mediante estandarización, los desarrolladores tanto de CBI (Computer-Based Instruction) como de ITS (Intelligent Tutoring Systems) centran su atención en:

Definir objetos educativos reutilizables.

Desarrollar nuevos modelos de contenidos

Desarrollar modelos de valoración de estudiantes.

Crear nuevos modelos de secuenciación de contenidos.

Crear bases de conocimiento educativo.

Cada uno de estos temas constituye un requisito para siguientes especificaciones de SCORM.

Actualmente la versión más extendida de SCORM es la 1.2, ya que la 2004 es demasiado reciente, y no existen demasiadas aplicaciones que la soporten, aunque se espera que en un futuro muy cercano la versión 2004 sustituya a la 1.2 y se convierta en el modelo a seguir por todos los desarrolladores de e-learning.

SCORM 2004 ha introducido varios cambios sobre versiones anteriores, en base a la evolución de los estándares en los que se basa.

SCORM 2004 se divide en 3 libros particulares, los cuales ADL pretende versionar y mantener individualmente. Cada uno tiene una cobertura determinada.

Run-Time Environment Book

Content Agreggation Model

Sequencing and Navegation

Por otro lado tenemos el libro “SCORM 2004 Overview”, que nos introduce en el qué y para qué de SCORM y “SCORM 2004 Addendum”, con las últimas modificaciones y errores que se han detectado en el estándar.

Estos documentos los podemos descargar de la página de http://www.adlnet.org.

#### 1.1.1.1.- Motivación del e-learning y SCORM

El empuje para la investigación en el área del e-learning viene promovido por la idea de que la gente aprenda más y mejor, reduciendo los costes en todo lo posible.

Estudios demuestran que:

- La velocidad con que los estudiantes pueden progresar en los estudios varía entre factores del 3% al 7%.
- De media, un estudiante en clase puede hacer 0,1 preguntas por hora
- En una tutoría personalizada este índice aumenta hasta 120 preguntas por hora.
- Gracias a las clases individuales se puede pasar de conseguir elevar el nivel del 50 % de alumnos a incluso un 98%.

Las clases individuales en ocasiones ofrecen los resultados ideales de la enseñanza. Sin embargo estas enseñanzas individuales serían imposibles de realizar a nivel logísticos y de costes si no fuera gracias al e-learning.

El objetivo de SCORM es conseguir que todo el mundo trabaje, en el campo del e-learning, de la misma forma, de modo que podamos conseguir que cualquier persona pueda crear un curso, adecuándose a unas características y restricciones que proporciona SCORM, de modo que ese curso pueda ser mostrado en cualquier LMS (Learning Management System) que haya sido diseñado también bajo el modelo de SCORM. Así conseguimos una completa independencia entre el curso y la plataforma donde se ejecuta.

### 1.1.1.2.- Intelligent Tutoring Systems (ITS)

Un sistema de enseñanza inteligente requiere:

Generar lecciones en tiempo real bajo demanda adecuadas para las características de un estudiante o usuario particular.

Tratar iniciativas de diálogo, intentando soportar un intercambio libre entre el estudiante y la tecnología.

Diversos factores perjudican el desarrollo de la tecnología ITS:

- La relativa inmadurez en nuestros días del estudio del conocimiento humano.
- El complejo modelado de la información y los sistemas basados en reglas requieren un gran poder de computación.

### 1.1.1.3.- Descripción del término LMS

Las siglas LMS (Learning Management System) son muy usadas en los documentos de SCORM.

Cuando hablamos de un LMS nos referimos al sistema diseñado para administrar cursos, mostrarlos a estudiantes particulares, almacenar el perfil de dichos estudiantes, etc. Es decir, nos referimos al sistema que se encuentra entre el navegador web de la persona que está viendo curso y el contenido educativo en sí (SCORM Content Package), especificado mediante las reglas que define SCORM.

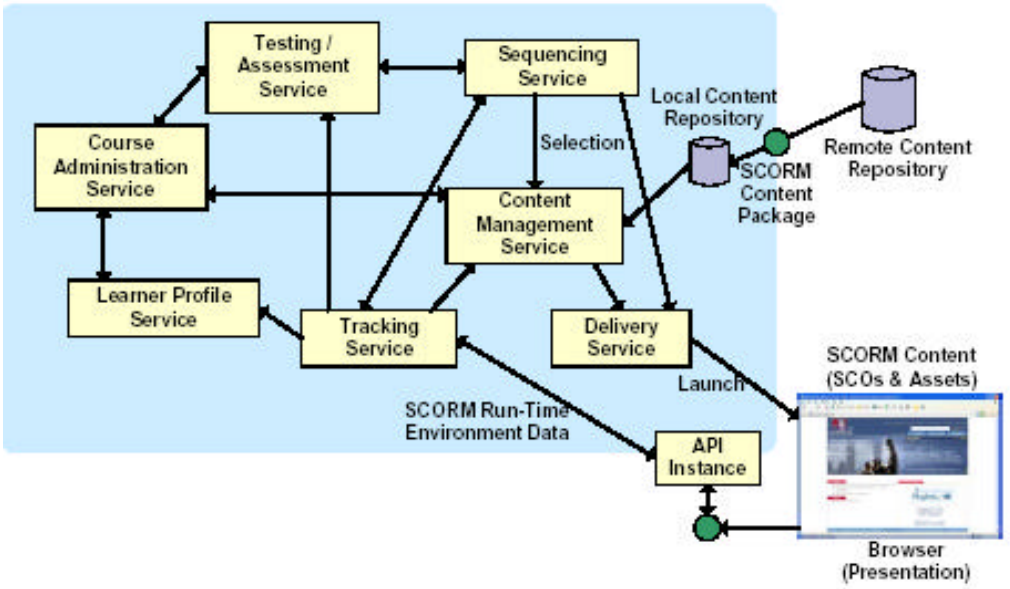



*Figure 1.5.3a: Highly generalized model of an LMS*

### 1.1.2.- CONTENT AGREGATION MODEL.

En este capítulo trataremos de describir el modelo de datos utilizado para diseñar los paquetes de contenidos.

Content Agregation Model representa un conjunto de reglas que deben ser seguidas a la hora de diseñar un paquete o recurso educativo (un curso, un tema, una parte de un curso, etc.) que pretenda ser mostrado a través de un LMS que siga el modelo SCORM 2004.

#### 1.1.2.1.- Partes del modelo de contenido.

Los componentes con los que se forman los paquetes de contenidos son dos, asset y Sco.

- *Asset*

Es la forma más básica de representación de conocimiento. Puede ser texto, sonido, imágenes, fragmentos de código, etc.

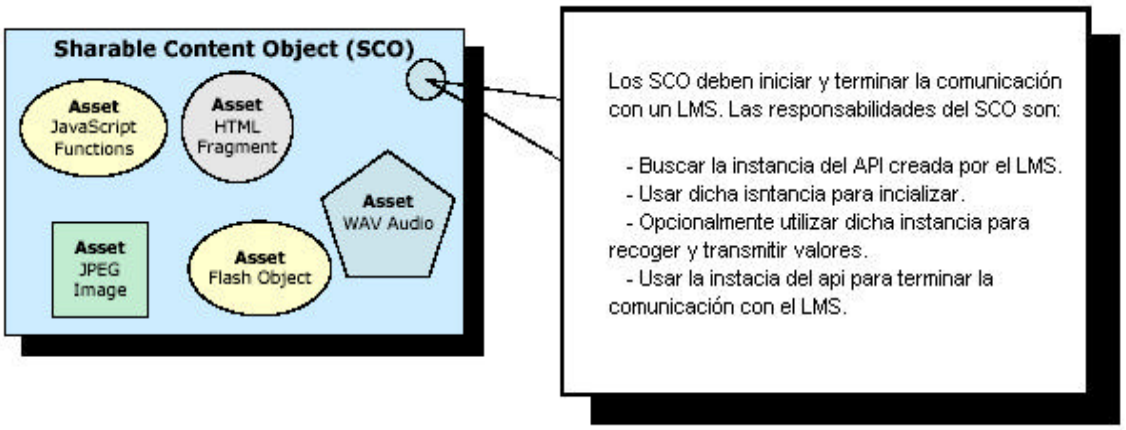
Un Asset puede ser descrito por un Asset Meta-Data, para poder ser más fácilmente encontrado y descrito en futuras ocasiones.

- *Sharable Content Objects (SCO)*

Un SCO es una colección de uno o más Assets que representa un único contenido ejecutable. SCO utiliza SCORM RTE para comunicarse con un LMS. Es el más bajo nivel de un contenido que puede ser manipulado mediante un LMS a través de RTE Data Model.

Debe ser independiente del contexto, ya que un objetivo importante es que un SCO sea fácilmente reutilizable.

Al igual que un Asset puede ser descrito mediante SCO Meta-Data.



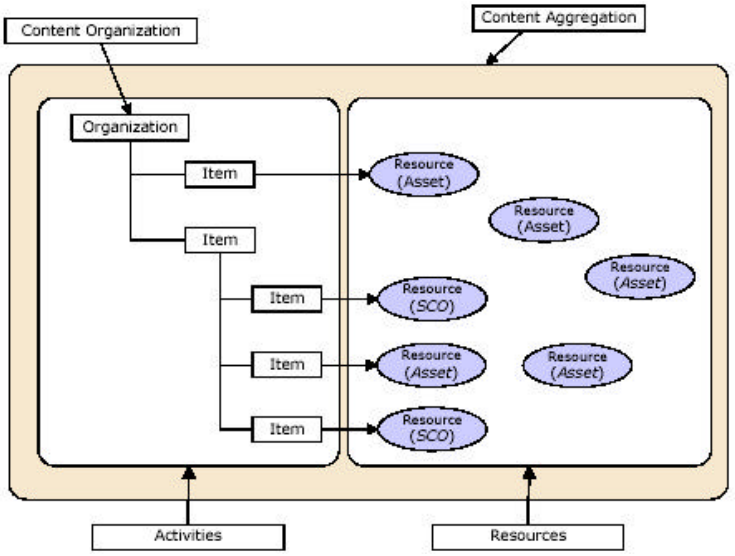
El hecho de que los SCO's deban comunicarse con el LMS a través de SCORM RTE beneficia la portabilidad, ya que:

- Cualquier LMS que soporte SCORM RTE puede lanzar SCO's, independientemente de quien lo haya creado.
- Cualquier LMS puede saber de cualquier SCO cuando ha sido inicializado y cuando finalizado.
- Un LMS que soporte SCORM RTE puede lanzar cualquier SCO del mismo modo.

*Content Organization*

La organización de el contenido se representa mediante una jerarquía de actividades. Las actividades pueden estar directamente asociadas con un contenido (Asset o SCO) que será utilizado para ejecutarla.

Las actividades que están formadas por otras actividades (niveles intermedios de la jerarquía) también son llamadas Clusters.




A través del Content organization Meta-Data podemos describir el Content Organization, lo que facilita la reutilización.

La secuenciación tan sólo se aplica a actividades, y es

independiente de los contenidos de los que conste el recurso, lo que es importante ya que la reutilización de un recurso está limitada a si tiene un contenido de secuenciación propio del contexto del curso. No obstante SCORM reconoce que en ocasiones es imposible evitar que un contenido tenga información de cómo debe tratarse dentro de un curso, es decir, información específica del contexto en el que se está usando.

*SCORM Meta-data*

SCORM recomienda el uso de LOM (Learning Object Metadata) del IEEE Learning Technology Standards Committee (LTSC) para la creación de Meta-data para cada uno de los componentes del modelo de contenido. El objetivo de estas descripciones no es otro que el de poder encontrar, categorizar y conocer más de cada uno de los componentes dentro de un sistema dado.

### 1.1.2.2.- Paquetes de contenido.

El IMS Content Packaging Specification fue diseñado para proporcionar un camino común a la hora de diseñar e intercambiar recursos educativos.

Los llamados Content packages tienen como finalidad el poder utilizar recursos educativos en distintos LMS´s.

SCORM Content Packages se adhiere estrictamente a el IMS Content Packaging Specification, y proporciona requisitos adicionales y detalles de implementación para el “empaquetado de Assets, SCO y Content Organization.

*Componentes de los paquetes de contenido.*

Existen dos componentes principales de un Content Package:

Un documento XML que describe de una forma estructurada el contenido del paquete e identifica los recursos de los que se compone. Es el llamado Manifiesto (imsmanifest.XML). Es necesario que este documento se localice en la ruta de la raíz del directorio del paquete.

Los ficheros físicos utilizados en la construcción del paquete.

Pueden utilizarse diferentes extensiones para definir la estructura del documento XML. Todos los archivos de control de la organización del XML (DTD, XSD, etc.) deben aparecer en la ruta del directorio del paquete, junto al imsmanifest.XML.

Los paquetes pueden ser comprimidos en formato zip o pif, de modo que se facilita su intercambio a través de la red entre los diferentes LMS. El archivo comprimido debe contener el manifiesto, los diferentes archivos de control del mismo y todos los ficheros que componen el paquete.

*Partes del manifiesto*

- Meta-Data: Descripción del paquete.

Organizations: Indica las diferentes formas de organización del paquete. Siempre debemos de tener una organización por defecto, pero puede haber varias organizaciones distintas dentro de un mismo manifiesto. Está información es utilizada por el LMS a la hora de mostrar el paquete al usuario.

Resources: Son los recursos de los que se compone el paquete (HTML, JavaScript, imágenes, etc).

Toda la información relativa a la construcción del manifiesto (etiquetas utilizadas, significado de las mismas, obligatoriedad, número, etc) la podemos encontrar en el libro SCORM Content Agregation Model versión 1.3.1, en el apartado 3.4.1.

### 1.1.3.- SCORM META-DATA

Resulta útil el poder describir los diferentes componentes del Content Package a la hora de buscar información por el sistema. Favorece claramente la reutilización e incluso se puede utilizar para, en tiempo de ejecución, seleccionar el componente que debe de ser mostrado al estudiante.

#### 1.1.3.1.- Construyendo SCORM Meta-data

El nodo raíz del Meta-Data debe de ser la etiqueta <LOM>, especificada por el IEEE para el diseño de metadatos en XML. Está definida en el espacio de nombres http://ltsc.ieee.org/xsd/LOM.

```
<lom xmlns="http://ltsc.ieee.org/xsd/LOM">
   <general/>
   <classification/>
   <annotation/>
   <lifeCycle/>
   <technical/>
   <metaMetadata/>
   <educational/>
   <relation/>
   <rights/>
</lom>
```

El nodo tiene nueve hijos, ninguno de los cuales es obligatorio. Además el orden de aparición de estos nodos hijo no es relevante. (SCORM Content Agregation Model v1.3.1 apartado 4.2 para detalles de cada uno de ellos)

Existen formas de validación de ficheros de metadatos diseñados mediante la etiqueta LOM. IEEE nos proporciona diferentes formas de validación de estos documentos XML mediante ficheros XSD. Tenemos validaciones estricta, media y baja.

Los metadatos pueden ser descritos directamente en el manifiesto o bien en archivos XML externos los cuales deberán de ser referenciados mediante la etiqueta <adlcp:location>.

```
<manifest>
   <metadata>
      <schema>ADL SCORM</schema>
      <schemaversion>CAM 1.3</schemaversion>
      <lom xmlns="http://ltsc.ieee.org/xsd/LOM">
         <general />
         <classification />
         <annotation />
         <lifeCycle />
         <technical />
         <metaMetadata />
         <educational />
         <relation />
         <rights />
      </lom>
   </metadata>
</manifest>
```

```
<manifest>
   <metadata>
      <schema>ADL SCORM</schema>
      <schemaversion>CAM 1.3</schemaversion>
      <adlcp:location>packageMetadata.xml</adlcp:location>
   </metadata>
</manifest>
```

Existen otras etiquetas en el manifiesto que son utilizadas para aportar información de cómo deben mostrarse los diferentes recursos educativos y características sobre la secuenciación de los mismo. Para conocerlos en detalle nos iremos al apartado 5 del Content Agregation Model de SCORM.

### 1.1.4.- SCORM RUN-TIME ENVIRONMENT

Dos metas de SCORM son que los paquetes de contenidos sean reutilizables e interoperables a través de múltiples LMSs. Para que esto sea posible debe haber una manera común de lanzar y de manejar paquetes de contenidos, un mecanismo común para que los paquetes de contenidos se comuniquen con un LMS y un vocabulario predefinido que forme la base de la comunicación. Estos tres aspectos del RTE son el lanzamiento, el interfaz de programación (API) y el modelo de datos.

El proceso del lanzamiento define una manera común para que LMSs ejecute objetos basados en la web. El proceso del lanzamiento define procedimientos y las responsabilidades del establecimiento de la comunicación entre el content object lanzado y el LMS. El mecanismo de comunicación se estandardiza con un API común.

El API es el mecanismo de la comunicación para relacionar al LMS con el content object. Se utiliza para recuperar y almacenar los datos (ej., cuenta, límites de tiempo, etc.) entre el LMS y el SCO.

### 1.1.4.1.- Modelo de tiempos

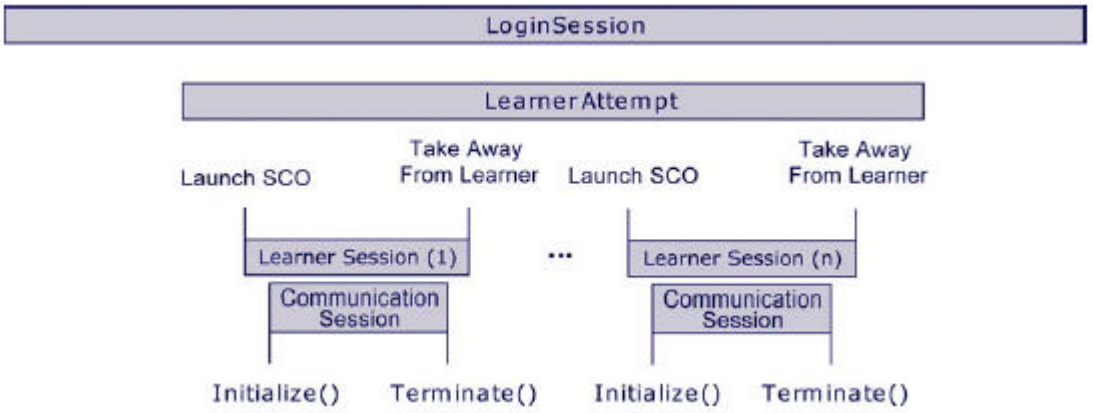
Varios aspectos claves necesitan ser definidos para ayudar en el seguimiento del estudiante durante la experiencia de aprendizaje.

Learner Attempt → Tiempo en el que un estudiante trata de superar los requisitos de una actividad que usa un content object. Un Attemp puede expandirse a lo largo de una o más learner Sessions, y puede ser suspendida entre lernear sessions.

Learner Session → Ininterrumpido periodo de tiempo durante el cual un estudiante está accediendo a un content object.

Communication Session → Comunicación activa entre un SCO y un Aplication Programing Interface (API)

Loggin Session → Periodo de tiempo entre que el estudiante inicia sesión en el sistema (logged on) y la termina (logged off).



### 1.1.4.2.- Lanzamiento de Content Objects

El proceso de lanzamiento define un método común para todo LMS de mostrar los Content Objects al navegador. Los procedimientos y responsabilidades para establecer esta comunicación varían dependiendo del tipo de Content Object que se lance (Asset o SCO).

- *Assets*

El modelo de lanzamiento de SCORM sólo requiere que el LMS lance el Asset utilizando el protocolo http. El Asset no se comunica con el LMS a través del API.

- *Sharable Content Objects (SCO´s)*

El modelo de lanzamiento de SCORM requiere que el LMS lance tan solo un SCO por estudiante. El SCO lanzado puede, a su vez, implementar una instancia del API y lanzar SCO´s subordinados a éste. El LMS no tiene por qué conocer de estos SCO´s subordinados, por lo que el SCO lanzado es el responsable de la limpieza de éstos (por ejemplo de cerrar las ventanas abiertas por los mismos).

El LMS debe lanzar el SCO en una ventana del navegador que sea dependiente o un frame hijo de la ventana del LMS que expone la instancia del API como un Document Object Model. El API Instance debe de ser proporcionado por el LMS.

Es responsabilidad del SCO buscar recursivamente a través de la jerarquía de frames padres y ventanas padres hasta encontrar la instancia del API. Una vez que lo encuentra el SCO puede iniciar la comunicación con el LMS.

### 1.1.5.- Aplication Programing Interface

Un API Implementation es una parte de software que implementa las funciones del API. Quien implemente las funciones del API no es de interés para el diseñador del SCO. El LMS necesita proporcionar una implementación del API que haga pública ésta interfaz a los SCO´s clientes.

Un API Instance es una ejecución individual de un API Implementation. Representa una porción de código en ejecución que interactúa con las operaciones del SCO.

El API proporciona un mecanismo de comunicación entre el SCO y el LMS. Toda comunicación entre el LMS y el SCO debe de ser iniciada por éste último. El SCO llama a las funciones definidas por el API Instance. El API Instance no invoca funciones definidas por el SCO.

#### 1.1.5.1.- Métodos de Sesión

El SCO utiliza métodos de sesión para iniciar y terminar la comunicación de datos con un API Instance.

- Initialize: Función utilizada para iniciar un Communication Session.

- Terminate: Utilizada para terminar un Communication Session. Es utilizada por el SCO cuando determina que no es necesario alargar más la comunicación con el LMS.

### 1.1.5.2.- Métodos de transferencia de datos

Un SCO usa estos métodos para enviar y recibir datos de un LMS dentro de un Communication Session. El LMS puede utilizar estos datos para tomar distintas decisiones.

- GetValue(parámetro). Permite a un SCO recoger información de un LMS. El parámetro identifica el tipo de dato requerido.

- SetValue(parámetro1,parámetro2). El método es utilizado para enviar datos a un LSM. Se envía el parámetro 2, que es del tipo de datos que indica el parámetro1. El API Instance puede determinar inmediatamente la persistencia de dicho parámetro, lo que indicará que debe de almacenarse directamente en el servidor o en la caché del navegador del cliente.

- Commit(parámetro). El método indica que deben de almacenarse todos los datos que hayan sido situados en la caché del navegador desde la última llamada a Initialize o Commit.Si no hubiera datos en caché el método no hará nada. El parámetro es la cadena vacía.

### 1.1.5.3.- Métodos de soporte.

GetLastError(). Devuelve el código del último error ocurrido.

GetErrorString(parámetro). Devuelve una descripción del código de error especificado en parámetro.

GetDiagnostic(parámetro). El LMS devuelve una información de diagnóstico definida a través de la instancia del API. Según el parámetro que le demos devolverá una u otra. El parámetro puede ser un código de error, pero también puede no serlo.

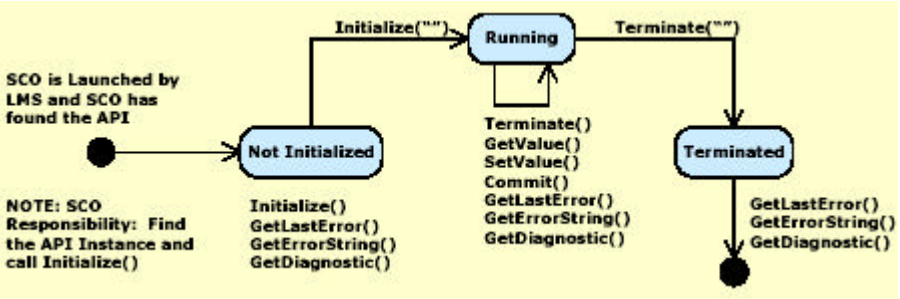
### 1.1.5.4.- Modelo de estados de un Communication Session

El IEEE define un modelo de estado conceptual para un API Instance durante su existencia. Tenemos tres estados:

No inicializado: El SCO ha sido lanzado, pero aún no se ha invocado la función Initialize(). El SCO debe buscar el API Instance proporcionado por el LMS.

Running: Desde que el SCO invoca Initialize() y hasta que llama a la función Terminate()

Terminated: El SCO ya ha invocado la función Terminate().




### 1.1.5.5.- Algunas reglas generales del API

Los nombres de las funciones son sensibles a las mayúsculas, y deben de ser definidos tal y como explica el Run-Time Environmet Book de SCORM

Los parámetros son sensibles a las mayúsculas. Todos los parámetros deben de ser representados en minúsculas.

Cada llamada a una función del API, excluyendo lo métodos de soporte, establece un código de error. (“0” en el caso de que no se produzca error).

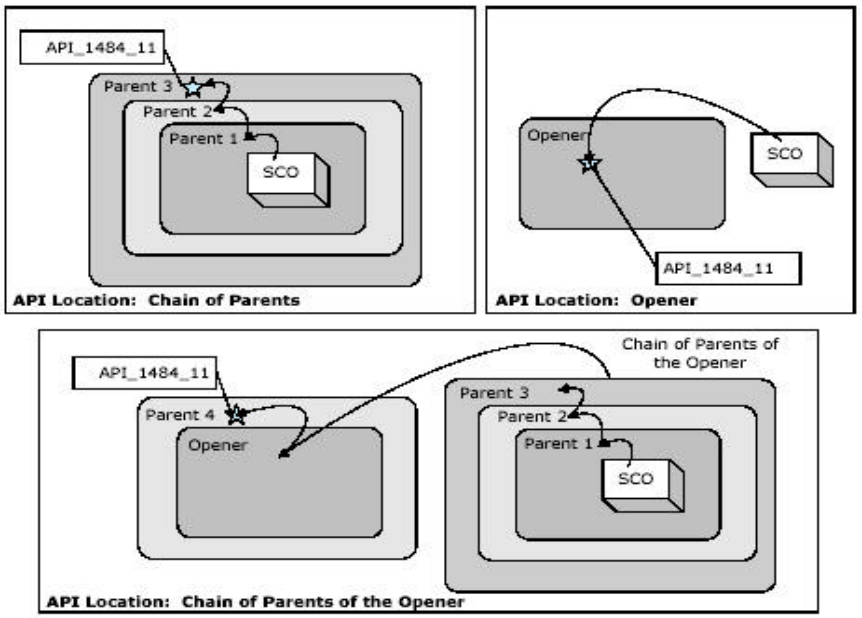
Todos los parámetros intercambiados entre un SCO y una instancia del API son tratados como ECMAScript strings (strings de JavaScript).

### 1.1.5.6.- Responsabilidades de un LMS

SCORM requiere que el LMS proporcione una instancia del API que implemente todas las funcionalidades de API de SCORM descritas en el Run-Time Environmet Book. El API Instance debe de ser accesible vía DOM, como un objeto llamado “API_1484_11”. El LMS debe de proporcionar la capacidad de encontrar una instancia del API al SCO mediante ECMAScript.

El LMS es el responsable de lanzar los SCO´s en una jerarquía particular DOM. Lanzará los SCO´s en una ventana del navegador que sea o bien ventana hijo o bien frame hijo de la ventana en la que se encuentra la instancia del API.

La siguiente imagen describe de forma muy visual el algoritmo especificado por el IEEE que debe de seguir un SCO para encontrar la instancia del API.




### 1.1.5.7.- Responsabilidades de un SCO

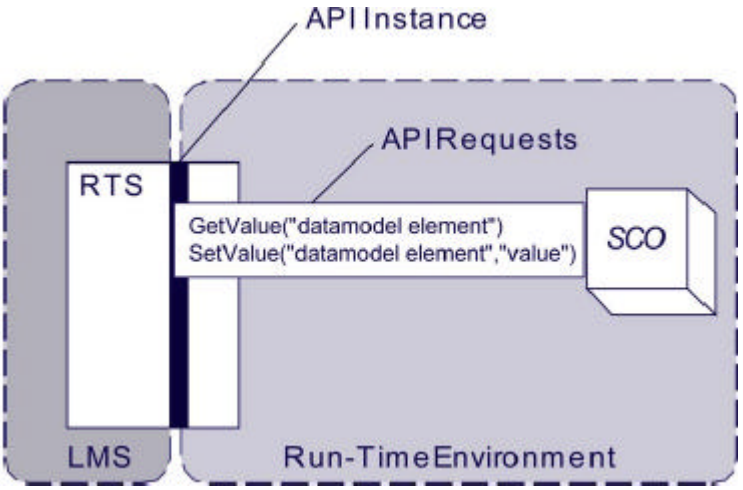
Todos los SCO´s tienen ciertas responsabilidades cuando se comunican a través de un API Instance.

El SCO debe buscar la instancia del API siguiendo el algoritmo que describe el apartado anterior y parar tan pronto como se encuentre. Una vez haya sido encontrado debe de invocar al menos dos funciones: Initialize() y Terminate(). El IEEE propociona un código que busca este API de forma consistente. No es necesario utilizar este código, puede definirse un equivalente.

```
var nFindAPITries = 0;
var API = null;
var maxTries = 500;
var APIVersion = "";

// The ScanForAPI() function searches for an object named API 1484 11
// in the window that is passed into the function.  If the object is
// found a reference to the object is returned to the calling function.
// If the instance is found the SCO now has a handle to the LMS
// provided API Instance.  The function searches a maximum number
// of parents of the current window.  If no object is found the
// function returns a null reference.  This function also reassigns a
// value to the win parameter passed in, based on the number of
// parents.  At the end of the function call, the win variable will be
// set to the upper most parent in the chain of parents.
function ScanForAPI(win)
{
   while ((win.API_1484_11 == null) && (win.parent != null)
          && (win.parent != win))
   {
      nFindAPITries++;
      if (nFindAPITries > maxTries)
      {
         return null;
      }
      win = win.parent;
   }
   return win.API_1484_11;
}

// The GetAPI() function begins the process of searching for the LMS
// provided API Instance.  The function takes in a parameter that
// represents the current window.  The function is built to search in a
// specific order and stop when the LMS provided API Instance is found.
// The function begins by searching the current window's parent, if the
// current window has a parent.  If the API Instance is not found, the
// function then checks to see if there are any opener windows.  If
// the window has an opener, the function begins to look for the
// API Instance in the opener window.
function GetAPI(win)
{
   if ((win.parent != null) && (win.parent != win))
   {
      API = ScanForAPI(win.parent);
   }
   if ((API == null) && (win.opener != null))
   {
      API = ScanForAPI(win.opener);
   }
   if (API != null)
      {
         APIVersion = API.version;
      }
   }
}
```

### 1.1.6.- SCORM RUN-TIME ENVIRONMENT DATA MODEL

El propósito de utilizar un modelo de datos común no es otro que el de que los SCO´s puedan ser utilizados por cualquier LMS que se adapte al modelo de SCORM.




Este conjunto de datos incluye información sobre los estudiantes, las lecciones, interacciones de los estudiantes con los SCO´s, objetivos, estado de satisfacción y completitud, etc. Esta información es muy valiosa a la hora de mostrar estadísticas del progreso del estudiante, tomar decisiones de secuenciación, etc.

#### 1.1.6.1.- Elementos del modelo de datos.

Todos los nombres de los elementos del modelo de datos descritos en el SCORM Run-Time Environment Data Model comienzan por cmi.

Los elementos del modelo de datos son utilizados opcionalmente por los SCO´s, ya que tan solo están obligados a invocar las funciones initialize y terminate, y no GetValue ni SetValue.

Podemos encontrar todo el modelo de datos en el libro Run-Time Environment Book de SCORM.

## 1.2.- LIP

### 1.2.1.- INTRODUCCIÓN

#### 1.2.1.1.- ¿Qué es LIP?

Las siglas LIP son el acrónimo de “Learner Information Package”. Tal y como su propio nombre indica, LIP trata del “Empaquetado de la información del alumno”. Esta especificación está basada en un modelo de datos que pretende describir aquéllas características de un alumno necesarias para:

- Registrar y manejar el historial, ambiciones y logros de los alumnos.
- Atraer a un alumno hacia la experiencia de aprender.
- Descubrir las oportunidades del aprendizaje para los alumnos.

El objetivo de LIP es, por tanto, proporcionar un medio de empaquetar la información del alumno de modo que los datos resultantes estén listos para ser intercambiados entre sistemas muy dispares (sistemas de información del alumno, SIA).

#### 1.2.1.2.- Principales características

- Información distribuida**:** Un sistema de información del alumno debe estar compuesto de múltiples sistemas distribuidos que comparten información del alumno. LIP es capaz de soportar el intercambio de datos entre SIAs distribuidos gracias a un sistema de referencia flexible usado para la identificación de los distintos registros y estructuras de datos encargados de almacenar la información del alumno. Existen dos mecanismos de referencia:

a. Sourcedid: Identifica el registro completo de información del alumno. El responsable de la creación del registro debe asegurarse de que cada expediente de la información del alumno tiene un identificador único. La unicidad de esta etiqueta está fuera del alcance de LIP y asume que cada servidor de información del alumno tiene una etiqueta única de la fuente pre-asignada a él. Una vez convenido el mecanismo para el "Identificador Global del Usuario" (GUID) se asume que ésta será la base para la generación del 'sourcedid' único.

b. Indexid: las once estructuras de datos base y las subestructuras asociadas usadas para contener la información del alumno pueden tener asignado un índice único dentro del registro de la información del alumno. Esto permite tener identificadas todas las categorías y, en caso de que haya modificaciones en el registro, no hace falta transferir todo el registro ya que tendríamos perfectamente identificadas las categorías que hubiesen cambiado. Una implicación importante es que el 'indexid' es un indicador persistente así que un sistema que lo utilice debe mantener una tabla mapeada entre el indexid (según lo utilizado para la interoperabilidad) y la estructura local de resolución de direcciones de la base de datos.

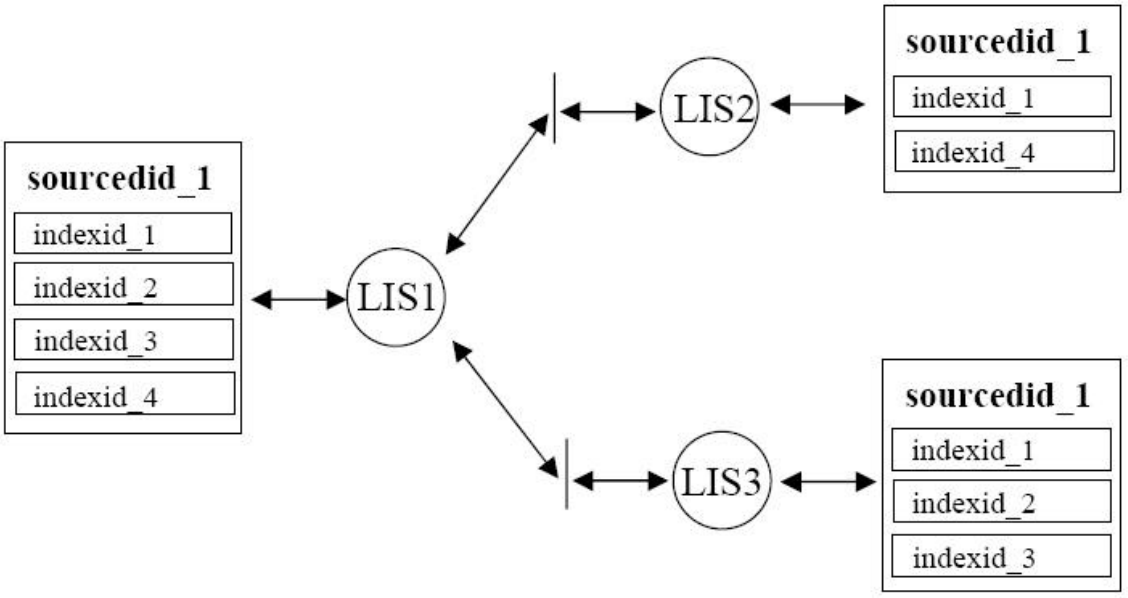



Figura 4. Referencia distribuida de la información del alumno

- Escalabilidad: Para soportar sistemas de gran escala es necesario intercambiar y reensamblar trozos de granularidad arbitraria.

  Los mecanismo usados por LIP para soportar la escalabilidad son:

✓ Granularidad de intercambio: esto se consigue mediante la división de la información del alumno en diversos paquetes claramente definidos, cada paquete es unívocamente identificado por lo que su reconstrucción puede llevarse a cabo sin problema.

✓ La estructura de intercambio puede consistir en una o más estructuras de información del alumno independientes o relacionadas. Esto significa que la transferencia de información puede beneficiarse de las estructuras de empaquetado más eficientes.

✓ Referencias externas a materiales a través del uso de URIs y XML.

- Privacidad y protección de datos: Los sistemas de información del alumno deben garantizar las políticas de privacidad y protección de datos y asegurar la integridad de dichos datos. LIP tiene dos mecanismos:

  ✓ Soporte para la inclusión de información que puede ser usada para describir el nivel de privacidad, derechos de acceso e integridad de los datos. Esto es definido como la meta-estructura de privacidad y protección de datos.

  ✓ Soporte para la información del alumno que puede ser usada para facilitar la transferencia de datos segura y/o autenticada (claves de seguridad).

- Flexibilidad y referencias externas: La información del alumno incluye muchos términos, como objetivos de aprendizaje e historial de aprendizaje, los cuales están representados por diferentas estructuras en diferentes contextos. Los modelos de datos de la información del alumno deben ser lo suficientemente flexibles como para acomodar esta necesidad.

### 1.2.1.3.- Estructura de los datos del alumno.

La información del alumno, tal y como muestra la figura 1, está dividida en once grandes categorías. Cabe destacar que la mayoría de elementos en LIP son opcionales lo cual hace que seamos nosotros quienes tengamos el control sobre la estructuración del empaquetado. Estas once categorías nos deben servir de guía para encontrar acomodo a nuestra información.

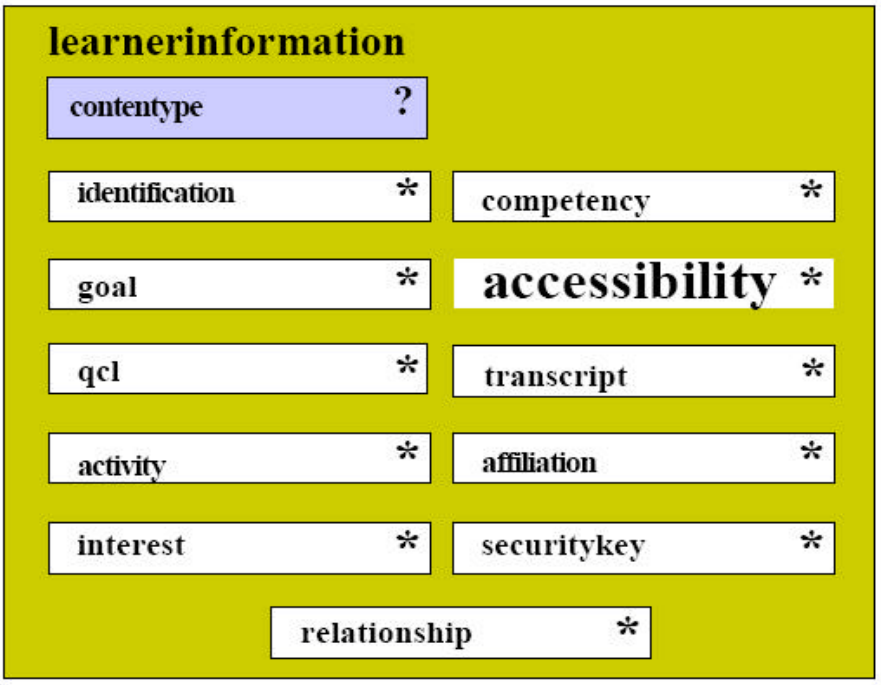



*Figura 1. La estructura de datos de IMS LIP*

1) **Identification (Identificación):** Datos biográficos y demográficos relevantes del alumno. También incluye información de contacto como, por ejemplo, el email.

2) **Goal (Propósito):** Área que empaqueta los logros personales y aspiraciones del alumno. Gracias a su estructura jerarquizada es muy fácil incluir sub-logros.

3) **Qcl (Qcl):** Refleja las calificaciones, certificados y diplomas obtenidos por el alumno así como la fuente del QCL y el nivel alcanzado.

4) **Activity (Actividad):** Cualquier actividad relacionada con el aprendizaje terminada o no. Incluyendo la auto-formación del alumno, la educación, tanto formal como informal, el entrenamiento, la experiencia laboral y el servicio militar o social.

5) **Transcript (Transcripción):** Expediente que se utiliza para proporcionar un resumen de los logros del alumno en una institución determinada. La estructura de este expediente puede tomar muchas formas.

6) **Interest (Intereses):** información sobre los hobbies y actividades de recreo. Estos ítems pueden estar relacionados con los datos de la categoría Qcl y pueden también contener representaciones digitales.

7) **Competency (Capacidades):** Aptitudes, conocimientos y habilidades adquiridas por el alumno tanto formal como informalmente. Estas habilidades pueden estar relacionadas con la información reflejada en las categorías *Activity* y/o *Qcl*.

8) **Affiliation (Afiliaciones):** este segmento se utiliza para tener información de todas aquellas organizaciones asociadas con el alumno. Puede incluir grupos de trabajo, clubs o asociaciones profesionales.

9) **Accesiblity (Accesibilidad):** Contiene las opciones de accesibilidad general a la información del alumno, es decir, las capacidades y/o incapacidades que posee el alumno para interactuar con el medio educativo.

10) **Securitykey (Llave de seguridad):** Conjunto de contraseñas y claves de seguridad asignadas al alumno para la transacción con los sistemas y servicios de la información del alumno.

11) **Relationship (Relaciones):** Ésta es un área importante que almacena el conjunto de relaciones existentes entre los anteriores diez componentes. Todas estas relaciones se capturan en una sola estructura de tal modo que hace los acoplamientos más simples de identificar y manejar como, por ejemplo, el borrado o actualizado de información.

LIP incorpora, para cada una de las categorías, tres campos comunes a todas ellas →

a) **Contentype:** este elemento contiene información:

- ✓ **Referencial** (Referencial): información que asigna un identificador único al dato componente.
- ✓ **Temporal** (Temporal): información que describe el desarrollo temporal de la información.
- ✓ **Privacy** (Privacidad): información que describe la privacidad y asegura la integridad de los datos.

b) **Comment**: esta área es la encargada de contener los comentarios pertinentes a la categoría en cuestión.

c) **Extension**: este es un campo *"preventivo"* para poder asumir toda clase de extensiones a la categoría.


Figura 2.

### 1.2.2.- SISTEMAS DE INFORMACIÓN DEL ALUMNO

#### 1.2.2.1.- Ejemplo de SIA

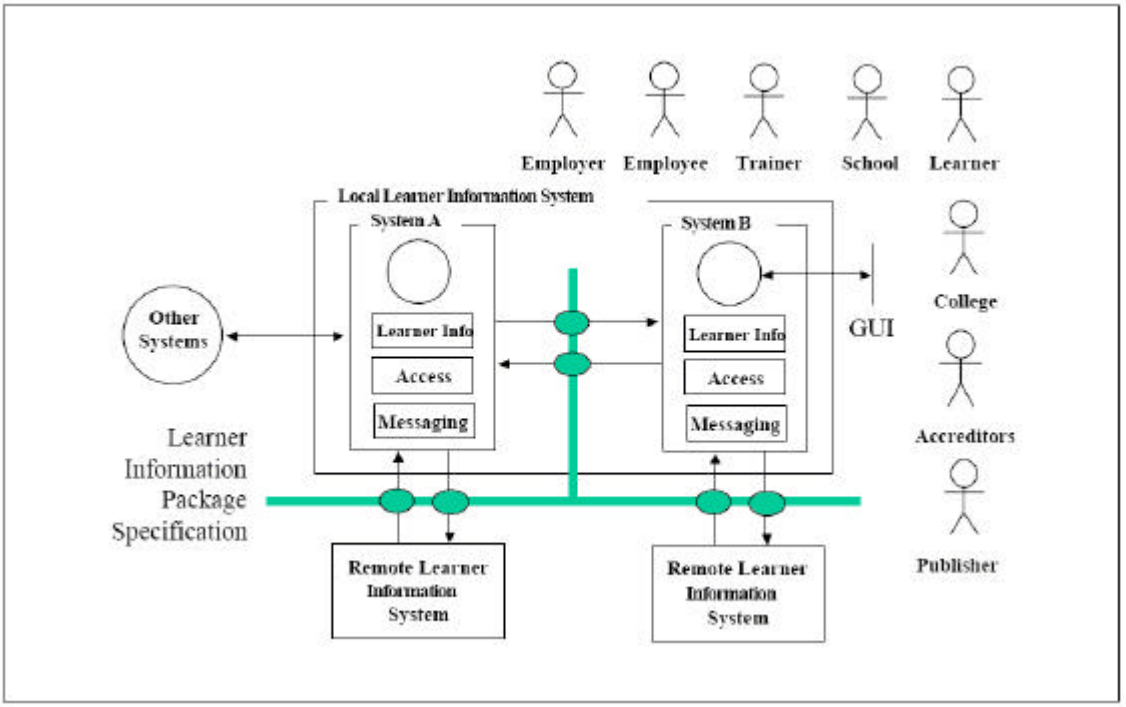



*Figura 3. Componentes de un SIA.*

- **Local Learner Information System** (Sistema local de información del alumno): los servidores locales que son directamente accesibles por el correspondiente usuario.

- **Remote Learner Information System** (Sistema remoto de información del alumno): un reflejo de la naturaleza distribuida de un servidor de información del alumno (diferentes partes de la información del alumno pueden ser almacenadas en varios servidores)

- **Other Systems** (Otros sistemas): otros sistemas que pueden ser interconectados a los servidores de información del alumno (p.e. e-mail) Los interfaces para estos sistemas están fuera del alcance de esta especificación.

- **Estructuras de datos:**
  - **Learner Info** (Información del alumno): datos actuales del alumno
  - **Access** (Acceso): derechos de acceso a los datos del alumno.
  - **Messaging** (Mensajería): protocolo de mensajería usado para implementar el actual intercambio de datos.

- **Actores:** Los diferentes roles de los usuarios que tienen acceso al servidor de datos. Los actores accederán al sistema a través de una 'GUI'.

#### 1.2.2.2.- Información del alumno

El IMS LIP fija su atención en la información del alumno. Las típicas clases de información del alumno soportadas son:

- Expediente educativo: el expediente de los logros educativos desde el colegio hasta la universidad. Los diferentes sistemas educativos en todo el mundo que hace falta soportar.

- Log de entrenamiento: el registro de las actividades de entrenamiento abordadas (p.e. cursos certificados)

- Registro del desarrollo profesional: el registro de las actividades profesionales desarrolladas incluyendo la asociación en los organismos apropiados profesionales.

- Currículum vitae: un registro de los logros personales incluyendo experiencia laboral, calificaciones e historial educativo.

- Registro de aprendizaje: registro sobre las actividades de aprendizaje y los logros de un individuo.

- Registro sobre los servicios a la comunidad.

### 1.2.2.3.- Categorías de la información del alumno

| | **Aprendices** | **Productores** |
|---|---|---|
| | Alumnos y grupos de alumnos. | Creadores<br>Proveedores<br>Vendedores |
| **Atributos fijos** | Identificación y localización.<br>Características físicas, técnicas y cognitivas. | Identificación y localización.<br>Tipo de organización. |
| **Atributos variables** | Objetivos, condiciones temporales, preferencias de aprendizaje. | Líneas de producción |
| **Portafolio personal** | Trabajos producidos<br>Referencias<br>Experiencia y educación | Trabajos de muestra<br>Profesionalidad alegada por los productores |
| **Portafolio de terceros** | Certificados | Calificaciones profesionales<br>Testimonios |

### 1.2.3.- CASOS DE USO

En la especificación original están presentes los siguientes ejemplos de casos de uso:

- Perspectiva individual: el proceso que sería emprendido por un individuo que está solicitando un trabajo y desea incluir su historial educativo y laboral en su aplicación.

- Perspectiva del proveedor: los procesos que deben ser soportados por los productos de los proveedores que desean proporcionar sistemas de gerencia de recursos humanos y sistemas de administración de estudiantes.

- Educación superior: información del alumno que debe ser intercambiada entre organizaciones (p.e. vía internet)

- Gestión de la carrera profesional: un registro con los logros de la carrera y planes asociados y monitorizados de los objetivos y ambiciones de la carrera.

- Gestión de los grupos de aprendizaje : la gestión de los grupos en proceso de aprendizaje, por ejemplo un grupo de empleados que están teniendo un aprendizaje inductivo.

### 1.2.4.- MODELO BÁSICO DE INFORMACIÓN

El modelo completo de información tiene tres componentes estrechamente relacionados, aunque esta especificación sólo se ocupa de la primera:

a. **Learner Information Package** (Paquete de la información del alumno): estructuras de datos base que contienen la información del alumno.

b. **Learner Information Exchange** (Intercambio de la información del alumno): soporte para la agregación, transacción, y protocolo de mensajes para los paquetes.

c. **Learner Information Query** (Consulta de la información del alumno): mecanismo de consulta por el que podemos recuperar la información del alumno dependiendo del criterio de búsqueda.

### 1.2.5.- REPRESENTACIÓN XML

El modelo de datos *Learner Information Packaging* ha sido definido como una jerarquía por lo que XML es perfecto para representar la información del alumno ya que XML es una jerarquía compuesta de elementos que tienen contenidos y atributos.

#### 1.2.5.1.- Elementos

Un elemento es un componente de un documento que ha sido identificado de un modo que una computadora lo puede entender. Cada elemento tiene una "etiqueta nombre" (tag name). Un elemento puede tener contenidos así como uno o mas atributos.

<TAGNAME>contents</TAGNAME>

**(a) Elementos. Contenidos.**

Un elemento puede contener otros elementos, *Parsed Charácter Data* (PCDATA), *Carácter Data* (CDATA), o una mezcla de PCDATA y elementos. Los contenidos aceptados de un elemento componen su modelo de contenido.

**(b) Elementos. Atributos.**

Un atributo proporciona información adicional acerca de un elemento. Los atributos son una forma de proporcionar características o propiedades a los elementos de un documento. Un elemento puede tener más de un atributo. Están contenidos dentro de la etiqueta de inicio de un elemento.

```
<timeframe>
    <begin restrict="1">1999-07-23</begin>
</timeframe>
```

**(c) Elementos. Nombres.**

Cada elemento tiene un nombre único. LIP posee las siguientes reglas en los nombres:

- ✓ Todos los nombres de las etiquetas seguirán las reglas de nombramiento de la especificación XML Versión 1.0.
- ✓ Los nombres que empiecen por "XML" no están permitidos.
- ✓ Todos los nombres de elementos y atributos deben ir en minúsculas.
- ✓ No deben incluir palabras reservadas (DOCTYPE, ELEMENT, ATTLIST, ENTITY).
- ✓ Los nombres de las etiquetas no pueden ser redefinidos con la excepción de aquellos que son usados para las extensiones.

### 1.2.5.2.- Esquema XML

La representación XML de la versión 1.0 de LIP se define mediante un 'Esquema XML'. Este esquema define elementos, su modelo de contenidos y atributos. También define el vocabulario estándar IMS. El 'Esquema XML' define los tipos de los elementos y los grupos de atributos de forma separada a los elementos. Esto persigue tres propósitos:

- ✓ Los nombres de los elementos son declarados antes de la definición de tipos para prevenir confusiones.
- ✓ La estructuración de los tipos se puede manejar con mayor eficacia, incluyendo el soporte para tipos derivados en un futuro cercano.
- ✓ Los atributos pueden ser manejados con mayor eficacia, particularmente cuando los mismo atributos son usados por unos pocos elementos.

### 1.2.5.3.- DTD

Los nombres de las etiquetas, el modelo de contenidos y los atributos de los elementos son definidos en la DTD. Este puede existir como un archivo externo o en un bloque de texto interno al documento XML.

#### (a) Declarando Contenidos

La información que especifica el orden y uso de los contenidos admitidos para un elemento constituye el modelo de contenido.

<!ELEMENT tagname (Content Model)> → <!ELEMENT short (#PCDATA)>

#### (b) Declarando Atributos

<!ELEMENT learnerinformation (sourcedid,goal?,transcript?,qcl?)>
<!ATTLIST learnerinformation type CDATA #IMPLIED>

La primera línea declara un elemento <learnerinformation> que debe tener el elemento <sourcedid> y además puede tener los elementos <goal> y/o <transcript> y/o <qcl> como su contenido. La segunda línea comienza con la palabra ATTLIST que sirve para iniciar una lista de declaración de atributos para el elemento <learnerinformation>. La palabra 'type' es el nombre del atributo cuyos valores serán del tipo CDATA. IMPLIED hace al atributo opcional.

### 1.2.5.4.- Extensibilidad

Para mejorar la extensibilidad, no debemos limitar posibles futuras extensiones en los elementos principales. Una extensión es el añadido de información a una estructura XML existente.

<!ELEMENT ext_qcl ANY>

Un ejemplo de extensión en el modelo de contenido del elemento <qcl> es:

<!ELEMENT qcl (title,registrationno,description,ext_qcl?)>

```
<qcl>
  <title>..</title>
  <registrationno>..</registrationno>
  <description>Text entry selections</description>
  <ext_qcl>
        <comments> This is a test to demo extensions </comments>
  </ext_qcl>
</qcl>
```

## 1.2.6.- DESCRIPCIÓN DE LA REPRESENTACIÓN XML

### 1.2.6.1.- <learnerinformation> Elementos

*Descripción:* El elemento <learnerinformation> es el contenedor más alejado de la información del alumno (es el LIP). Esta información puede ser acerca de un individuo o una organización.

*Multiplicidad:* El elemento <learnerinformation> sólo aparece una vez en cada archivo XML.

*Atributos:* XML:lang (opcional – default='en'). Identifica el lenguaje usado.

*Elementos:*

- ✓ <comment>: contiene los comentarios relevantes. 0 ó 1 veces.

  XML:lang(opcional – default='en')

- ✓ <contentype>: contiene la descripción del contenido de meta-datos concerniente al ńdice de los datos, derechos de acceso y estampación de tiempo. 0 ó 1 veces.

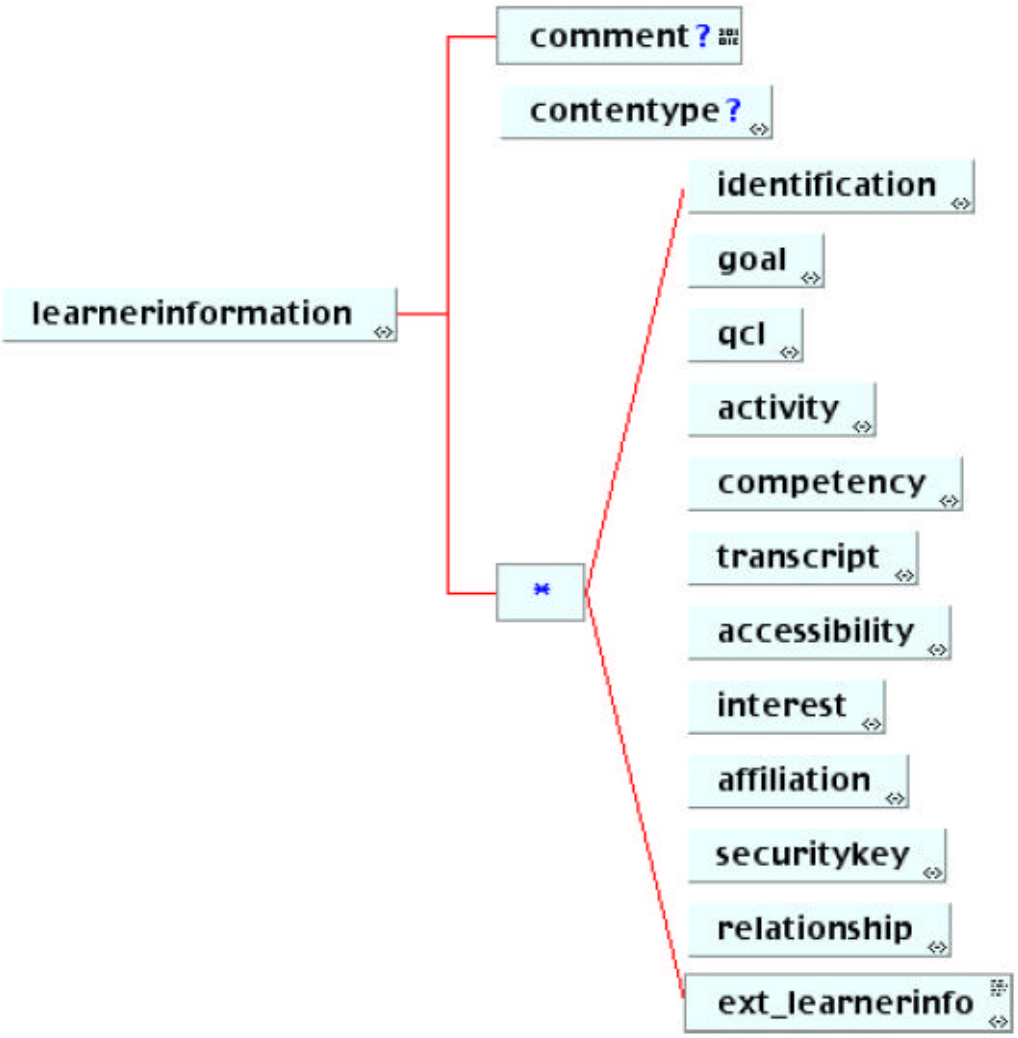



Figura 5. Elementos de <learnerinformation>

- ✓ **<identification>**
- ✓ **<goal>**
- ✓ **<qcl>**
- ✓ **<activity>**
- ✓ **<competency>**
- ✓ **<transcript>**
- ✓ **<accesility>**
- ✓ **<interest>**
- ✓ **<afiliation>**
- ✓ **<securitykey>**
- ✓ **<relationship>**
- ✓ **<ext_learnerinfo>**

La especificación de éstos elementos puede encontrarse de manera detallada en el documento original de LIP (XML Binding).

### 1.2.7.- RELACIONES CON OTRAS ESPECIFICACIONES

#### *1.2.7.1.-* Especificaciones IMS

La versión 1.0 de la especificación del IMS LIP consta de tres documentos:

- IMS Learner Information Package. Especificación del Modelo de Información – Versión 1.0 → Este documento describe la estructura de los datos que debe usarse para proporcionar interoperabilidad.

- IMS Learner Information Package. Ligadura XML – Versión 1.0 → Este documento describe como codificar los objetos de información del alumno en XML y proporcionar los correspondientes esquemas XML.

- IMS Learner Information Package. Mejor Práctica & Guía de Implementación – Versión 1.0 → Este documento proporciona una vista general y describe como el Modelo de Información IMS LIP y la Ligadura XML pueden ser aplicados a tipos específicos de escenarios de interoperabilidad.

La especificación IMS LIP está relacionada con algunas otras especificaciones IMS:

- IMS Meta-data Specification → recomienda el uso de las definiciones de meta-datos IMS para soportar las entidades de meta-datos que son usadas en el contexto de LIP.

- IMS Enterprise Specification → el IMS LIP no debe ser utilizado para el intercambio de la información referente grupos de la educación y a la calidad de miembro de tales grupos. Este tipo de intercambio de información se debe basar en el IMS Enterprise Specification;

- IMS Content Packaging Specification → múltiples instancias XML IMS LIP deben ser empaquetadas usando mecanismos descritos en este documento.

- IMS Question & Test Interoperability → IMS LIP es más general que QTI y por ello es capaz de soportar el método QTI.

### 1.2.7.2.- Especificaciones y actividades relacionadas

- Especificaciones:

- IEEE P1484
- Schools Interoperatibility Framework (SIF)
- ANSI TS 130 Student Educational Record
- Internet vCard Specification

- Internet2 eduPerson
- HR-XML Consortium Specifications

- Actividades:

- ISO/IEC JTCI/SC36 Learning Technology
- Advanced Distributed Learning (ADL) Iniciative
- Aviation Industry CBT Committee (AICC)
- EU ACTS Gestalt Project
- CEN/ISSS

### 1.2.7.3.- Especificación IMS. Proceso de desarrollo.

Desarrollo de una especificación IMS:

- Mes 1: creación del equipo incluyendo los líderes de proyecto, editores y grupos de colaboración.

- Mes 2: Desarrollo del equipo y del ámbito de requisitos.

- Mes 3: documentos iniciales desarrollados.

- Mes 4: Desarrollo del documento base y aprobación del mismo.

- Mes 5: Mejora del documento. Se identifican las ediciones abiertas y se desarrollan las soluciones. Se anima a las compañías que desarrollen código contra los documentos base.

- Mes 6: Mejora y regeneración adicionales del documento por parte de organizaciones implicadas en implementaciones.

- Mes 7: Terminación y aprobación del Public Draft Specification.

- Meses 8 y 9: Aceptación de las regeneraciones de las organizaciones que trabajan para el Public Draft Specification. Resolver cualquier asunto planteado.

- Mes 10: Terminar y aprobar la especificación final.

### 1.2.8.- ACCLIP

#### 1.2.8.1.- Introducción

La especificación ACCLIP añade modificaciones sobre el elemento <accesibility> que permite al alumno definir sus preferencias y explicar la interfaz y el uso de su terminal al utilizar el sistema de aprendizaje.

El usuario, al ir definiendo sus preferencias, se va creando un contexto (context).

Las preferencias del alumno están agrupadas en 3 grupos:

- <display> : define cómo quiere el alumno que se le presente la información en la pantalla.

- <control> : define cómo quiere el alumno controlar el dispositivo.

- <content> : define el contenido requerido por el alumno.

Cuando la especificación de LIP se publicó en Marzo de 2001, se dejó sitio para la información referente a los usuarios con discapacidades. El ACCLIP completa ese sitio y proporciona una estructura para la información referente a las necesidades y preferencias tanto de los alumnos con discapacidades como para las de otros alumnos.

Modelo de la información de ACCLIP:

- Nos proporciona un medio para apoyar a los alumnos con o sin discapacidades con distintos valores de preferencia para contenidos e interfaces.

- Expresa necesidades de accesibilidad y permitirá al sistema de aprendizaje localizar y presentar el material que se necesite con respecto a las necesidades expuestas.

- Define la manera que tiene el usuario de interactuar con el sistema (sólo con teclado, sólo con ratón, o pantalla táctil).

- Permite al usuario especificar el tipo de material apropiado para sus necesidades, por ejemplo si necesita leyendas o descripciones a través de ficheros de audio.

- Servirá para las personas con muchos tipos de problemas de acceso. Estos tipos son variados, algunos de los que influyen son los siguientes:

    - bajo ancho de banda
    - hardware viejo o lento
    - falta de hardware
    - uso de PDA u otro dispositivo (teléfono)
    - entorno ruidoso o tranquilo
    - multitarea

Estas modificaciones en la especificación de IMS evita muchos problemas para los usuarios con discapacidades permitiendo al sistema adaptar sus pantallas, control y características de contenido según las necesidades del usuario.

### 1.2.8.2.- AccessForAll

La etiqueta *accesForAll* de acclip equivale a *disability* del estándar LIP versión 1.0. Dicha etiqueta *disability* ha sido eliminada del estándar.

*AccesForAll* puede aparecer ninguna, una o varias veces, conteniendo obligatoriamente la etiqueta hijo *context,* que podrá aparecer varias veces.

**(a) Context**

Cada contenido de la etiqueta define un conjunto de características de preferencias de acceso del usuario. De este modo podemos hacer constar en los sistemas las preferencias de los usuarios en cuanto al acceso a la aplicación.

*Context* contiene tres etiquetas hijo, cada una de las cuales define una característica particular.

**(b) Display**

Esta etiqueta permite definir preferencias de cómo se muestra o se comunica el material con el alumno.

Todas las etiquetas hijo de <display> son optativas y pueden aparecer tan sólo una vez, excepto <futureTechonology> que podría aparecer un número indefinido de veces.


- **ScreemReader**: Tecnología que muestra el texto a través de un sintetizador de discurso.

- **ScreemEnhance**: Tecnología que muestra el texto para que sea más fácil de ver. Por ejemplo, aumentar el tamaño de la fuente, poner mayor contraste, cambiar el tipo de letra, cambiar el color de fondo, cambiar el tamaño del cursor...

- **TextReadingHighlight**: Destacar como se lee el texto en un sintetizador de discurso. Se puede definir las palabras por minuto que se leen, el volumen de la voz, si se lee un texto alternativo...

- **Braille**: Utilizado cuando un dispositivo muestra el texto y otra información usando Braille. Se puede definir el grado de braille, el número de puntos, si se marca el texto destacado, si se marca el texto en negrita, si se marca el texto subrayado, si se marca el color,....

- **Tactile**: Tecnología que usa el tacto para interpretar la información.

- **VisualAlert**: Tecnología que nos proporciona alertas visuales para las alertas sonoras. Proporcionan estas alertas a través de destellos del escritorio o de la ventana activa o de la barra de título, proporciona subtítulos para el audio generado por el sistema...
- **StructuralPresentation**: Establece como se estructura el contenido que se muestra. Se puede indicar con que detalle se muestra la información en cada momento, si muestra imágenes o texto, si muestra una lista separada con los links existentes...

- **FutureTechnology**: Permite la extensibilidad. Podemos declarar características de tecnologías futuras.

**(c) Control**

A través de la etiqueta <control> podemos detallar las necesidades y preferencias del alumno a la hora de interactuar con el sistema. Podemos definir diferentes funcionalidades de los dispositivos que el usuario utilizará para comunicarse.

Todas las etiquetas hijo de <control> son optativas y pueden aparecer tan sólo una vez, excepto <futureTechnology> que podría aparecer un número indefinido de veces.

- **KeyboardEnhanced**: define las características a destacar a la hora de comunicarse, el estudiante, con el sistema a través de un teclado estándar.

- **OnscreenKeyboard**: teclado virtual mostrado en la pantalla. En ella de se pueden definir las características preferidas por el usuario que tendrá dicho teclado.

- **AlternativeKeyboard:** a través de esta etiqueta nos estamos refiriendo a un hardware que funciona como un teclado genérico, pero en realidad es un dispositivo externo al sistema.

- **MouseEmulation**: formas alternativas al ratón. Métodos a través de los cuales podemos sustituir las funciones de ratón a través del teclado, reconocimiento de voz o cualquier otro dispositivo no señalador.

- **AlternativePointing**: tecnología que podríamos utilizar para sustituir un ratón por un dispositivo señalador. Detalles y configuraciones necesarias del mismo.

- **VoiceRecognition**: detalles de control para el reconocimiento de comandos de voz.

- **CodedInput**: Métodos de control que utilizan un código para seleccionar la entrada deseada.

- **Prediction**: Características de control a través de las cuales el sistema predice y/o completa la entrada del usuario.

- **StructuralNavigation**: Opciones para definir el control de la navegación.

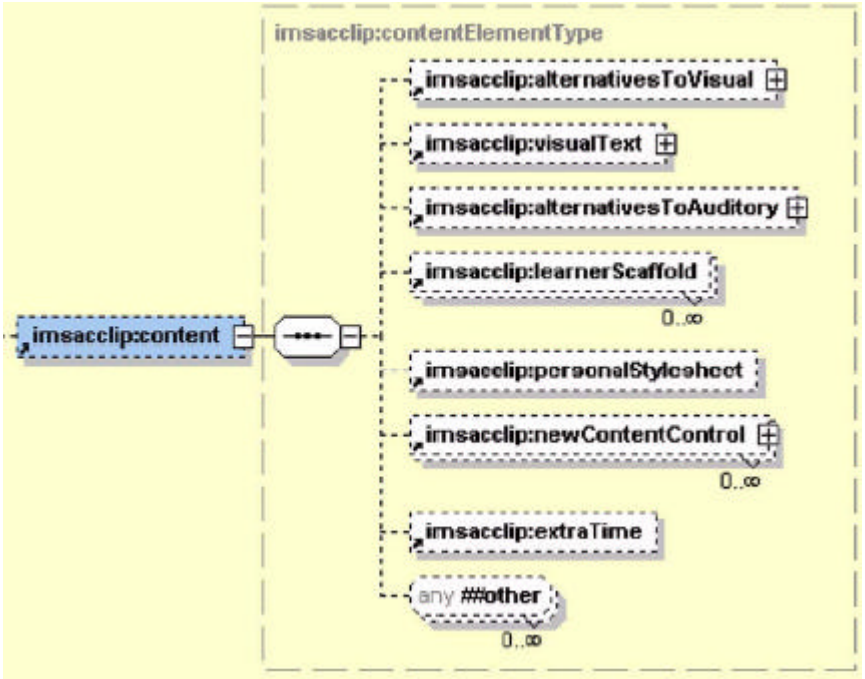
- **FutureTechnology**: Permite la extensión del estándar. Usado para declarar características necesarias para las futuras tecnologías que pudieran darse a conocer.

### (d) Content

Preferencias con respecto al contenido, especificando cualquier transformación o aspectos destacables.

A través de este elemento podemos definir atributos y características del contenido. Normalmente estas características estarán relacionadas con la información que aparecerá en los meta-data asociados al contenido. No ayudará a realizar búsquedas en los contenidos teniendo en cuenta las características relativas a la accesibilidad del usuario.

Todas las etiquetas hijo de <content> son optativas y pueden aparecer tan sólo una vez, excepto <learnerScaffold> y <futureTechnology> que podrían aparecer un número indefinido de veces.



- **AlternativesToVisual**: Características preferidas por el usuario a la hora de presentar los elementos visuales (a través de audio, altos contrastes, largas descripciones, etc).

- **AlternativesToText**: Características preferidas por el usuario cuando se debe presentar un contenido textual.

- **AlternativesToAuditory**: Formas de presentar el contenido auditivo. Podemos introducir subtítulos, definir la velocidad del audio, etc.

- **LearnerScaffold**: Con dicha etiqueta podemos hacer constar que herramientas son de uso común para el usuario (diccionario, calculadora, bloc de notas, etc). Esta etiqueta es múltiple, con lo que podemos hacerla aparecer las veces que sea necesaria. Una por cada herramienta que deseemos.

- **ExtraTime**: Permite que el usuario pueda pedir un tiempo extra a la hora de ver los contenido o de contestar a los requerimientos del sistema, por ejemplo, durante una prueba.

- **FutureTechnology**: Permite la extensibilidad. Podemos declarar características de tecnologías futuras.

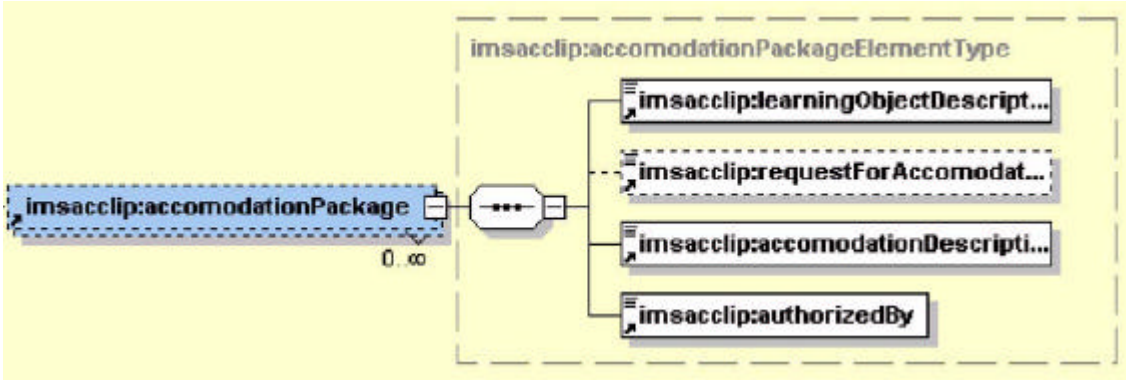
#### 1.2.8.3.- Cambios en <elegibility>

Se ha definido el elemento <accommodation> como una extensión de <elegibility>, que ya aparecía previamente en la definición de IMS LIP.

<accommodation> es utilizado para especificar las comodidades a las que puede acceder un usuario a la hora de afrontar un componente educativo. Es importante que una característica pensada para la comodidad del usuario no

interfiera en el propósito previsto para que el estudiante aprenda algo. Por ello dichas facilidades deben de ser aprobadas previamente por un administrador del sistema o profesor.

El elemento <accommodation>, contenido dentro de <elegibility>, contiene un número indefinido de "accommodation packages", que contienen informaciones tales como descripción del objeto educativo, comodidades permitidas para el usuario con dicho objeto educativo, quién autorizó dicha facilidad, los requisitos de ésta, etc.




### 1.2.8.4.- Ejemplos de uso

Los siguientes ejemplos muestran como se usa el XML para expresar preferencias de distinto tipo.

**(a) Ejemplo de preferencias de Display**

ScreemReader

Usamos esta etiqueta cuando el texto que aparece en la pantalla es leído usando un sintetizador de discurso.

En el ejemplo que vamos a ver el lector de pantalla tiene que leer los links que aparezcan en la ventana(link). La velocidad tiene que ser de 180 palabras por minuto (speechRate). El tono de voz se pone a su valor por defecto (pitch). También requiere un subir un poco el volumen (volume).

```
<screenReaderGeneric>
  <link usage="required" value="speakLink"/>
  <speechRate usage="preferred" value="180"/>
  <pitch usage="optionallyUse" value="0.5"/>
  <volume usage="required" value="0.5"/>
</screenReaderGeneric>
```

Braille

Algunas personas ciegas prefieren usar pantallas con braille. Estas pantallas tienen una serie de celdas con puntos que se levantan o se bajan.

En el siguiente ejemplo el alumno requiere braille descomprimido(grade). Cada celda tiene 8 puntos (numDots), y su pantalla tiene 80 celdas para cada fila (numCells). El alumno prefiere conocer las características del texto, y que opcionalmente indique cuando esta el texto destacado(markHighlight), negrita (markBold), subrayado(markUnderline), en cursiva (markItalic) y color (markColor). Prefiere la presión del punto de 0.5(dotPressure). Por último prefiere usar Braille Status Cell, con una celda extra usada para mostrar el estado del sistema(statusCell).

```
<brailleGeneric>
  <grade usage="required" value="1"/>
  <numDots usage="preferred" value="8"/>
  <numCells usage="preferred" value="80"/>
  <markHighlight usage="optionallyUse" value="true"/>
  <markBold usage=" optionallyUse " value="true"/>
  <markUnderline usage=" optionallyUse " value="true"/>
  <markItalic usage=" optionallyUse " value="true"/>
```

```
<markStrikeout usage=" optionallyUse " value="true"/>
<markColor usage=" optionallyUse " value="true"/>
<dotPressure usage="preferred" value="0.5"/>
<statusCell usage="preferred" value="left"/>
</brailleGeneric>
```

**(b) Ejemplos de preferencia de Display**

KeyBoardEnhanced

En el siguiente ejemplo el alumno prefiere usar una disposición de teclado externa que se encuentra en un archivo llamado keyboard.XML (loyout). El alumno prefiere no usar las “sticky keys”, las cuales deja presionada la techa de shift o la de control hasta que la siguiente tecla es presionada (stikyKeys). Prefiere no tener una tecla repetida si se deja pulsada, el retraso automático y el índice de repeticiones es un valor normal (repeatKey). El usuario tiene manos inestables y puede pulsar varias teclas a la vez sin querer. Por lo tanto hace falta quitar el rebote de las teclas(debounce).

```
<keyboardEnhancedGeneric>
        <layout>
                <alphaLayoutExternal value="keyboard.XML"/>
        </layout>
        <stickyKeys usage="required" value="false">
                <playSound usage="required" value="false"/>
        </stickyKeys>
        <repeatKeys usage="required" value="false">
                <autoRepeatDelay usage="required" value="0.5"/>
                <autoRepeatRate usage="required" value="0.5"/>
        </repeatKeys>
        <debounce usage="required" value="true">
                <debounceInterval usage="required" value="0.5"/>
        </debounce>
</keyboardEnhancedGeneric>
```

StructuralNavigation

Esta etiqueta se usa para poner los contenidos de navegación. En este caso el alumno prefiere navegar por el contenido en una profundidad de primer orden (navigationDepth) y utilizar una tabla de contenido(useTableOfContents).

```
<structuralNavigation>
    <navigationDepth usage="preferred" value="depthFirst"/>
    <useTableOfContents usage="preferred" value="true"/>
</structuralNavigation>
```

**(c) Ejemplos de preferencia de Control**

AlternativesToVisual

A menudo el contenido se hace disponible de distintas formas para que el usuario pueda elegir. Proporciona las preferencias para la presentación no-visual del contenido que se presenta normalmente de manera visual.

En este ejemplo el alumno prefiere usar una descripción de audio en inglés con la información ampliada, si esta disponible. Se debe evitar las combinaciones de rojo y verde (daltonismo).

```
<alternativesToVisual>
    <audioDescription usage="required" lang="en" type="expanded"/>
    <altTextLang usage="required" lang="en-us"/>
    <longDescriptionLang usage="required" lang="en-us"/>
    <colorAvoidance>
        <avoidBlueYellow usage="required" value="false"/>
        <avoidGreenYellow usage="required" value="false"/>
        <avoidOrange usage="required" value="false"/>
        <avoidPurpleGray usage="required" value="false"/>
        <avoidRed usage="required" value="false"/>
        <avoidRedBlack usage="required" value="false"/>
        <avoidRedGreen usage="required" value="true"/>
```

<useMaximumContrastMonochrome usage="required" value="false"/>
</colorAvoidance>
</alternativesToVisual>

#### 1.2.8.5.- PersonalStylesheet

Una hoja de estilo da al usuario un gran control sobre como se presenta el contenido. En estas hojas se puede definir el tamaño de la letra, el color, incluso la disposición puede ser definida y controlada.

<personalStylesheet usage="preferred" value="sheet.css"/>

# 2.- Fundamentos tecnológicos.

## 2.1.- J2EE

**J2EE** son las siglas de Java 2 Enterprise Edition. Es la edición empresarial del paquete Java creada y distribuida por Sun Microsystems (http://www.sun.com/). Comprenden un conjunto de especificaciones y funcionalidades orientadas al desarrollo de aplicaciones empresariales.

La plataforma J2EE se ha convertido en un *estándar de facto* para el desarrollo e implementación de aplicaciones habilitadas para operar en la Web y centradas en el servidor porque:

- Está basada en componentes estandarizados y modulares.
- Proporciona un conjunto completo de servicios a dichos componentes.
- Se hace cargo automáticamente de muchos detalles del comportamiento de las aplicaciones, sin necesidad de una compleja programación.

## 2.2.- XML

¿El objetivo? Distanciar definitivamente el contenido de su representación.

Las tecnologías XML se han mostrado como el camino idóneo, o al menos una gran ayuda, para representar información sin ligarla de ningún modo con una representación concreta.

**XML** (e**X**tensible **M**arkup **L**anguage) (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C).

Es una versión simple de SGML. Su objetivo principal es conseguir una página Web más semántica. Aunque una de las principales funciones con las que nace sería suceder al HTML, separando la estructura del contenido y permitiendo el desarrollo de vocabularios modulares, compatibles con cierta unidad y simplicidad del lenguaje (objetivo que se viene desarrollando a través de la especificación XHTML), tiene otras aplicaciones entre las que destaca su uso como estándar para el intercambio de datos entre diversas aplicaciones.

Al igual que el HTML, se basa en documentos de texto plano en los que se utilizan etiquetas para delimitar los elementos de un documento. Sin embargo, XML define estas etiquetas en función del tipo de datos que está describiendo y no de la apariencia final que tendrán en pantalla o en la copia impresa, además de permitir definir nuevas etiquetas y ampliar las existentes.

**XSL** (e**X**tensible **S**tylesheet **L**anguage) es una familia de lenguajes basados en el estándar XML que permite describir cómo la información contenida en un documento XML cualquiera debe ser transformada o formateada para su presentación en un medio específico.

Un caso particular de XSL son las **XSLT** (e**X**tensible **S**tylesheet **L**anguage **T**ransformations), que permite convertir documentos XML de una sintaxis a otra

(por ejemplo, de un XML a otro, a un documento HTML o a un documento PDF).

## 2.3.- STRUTS

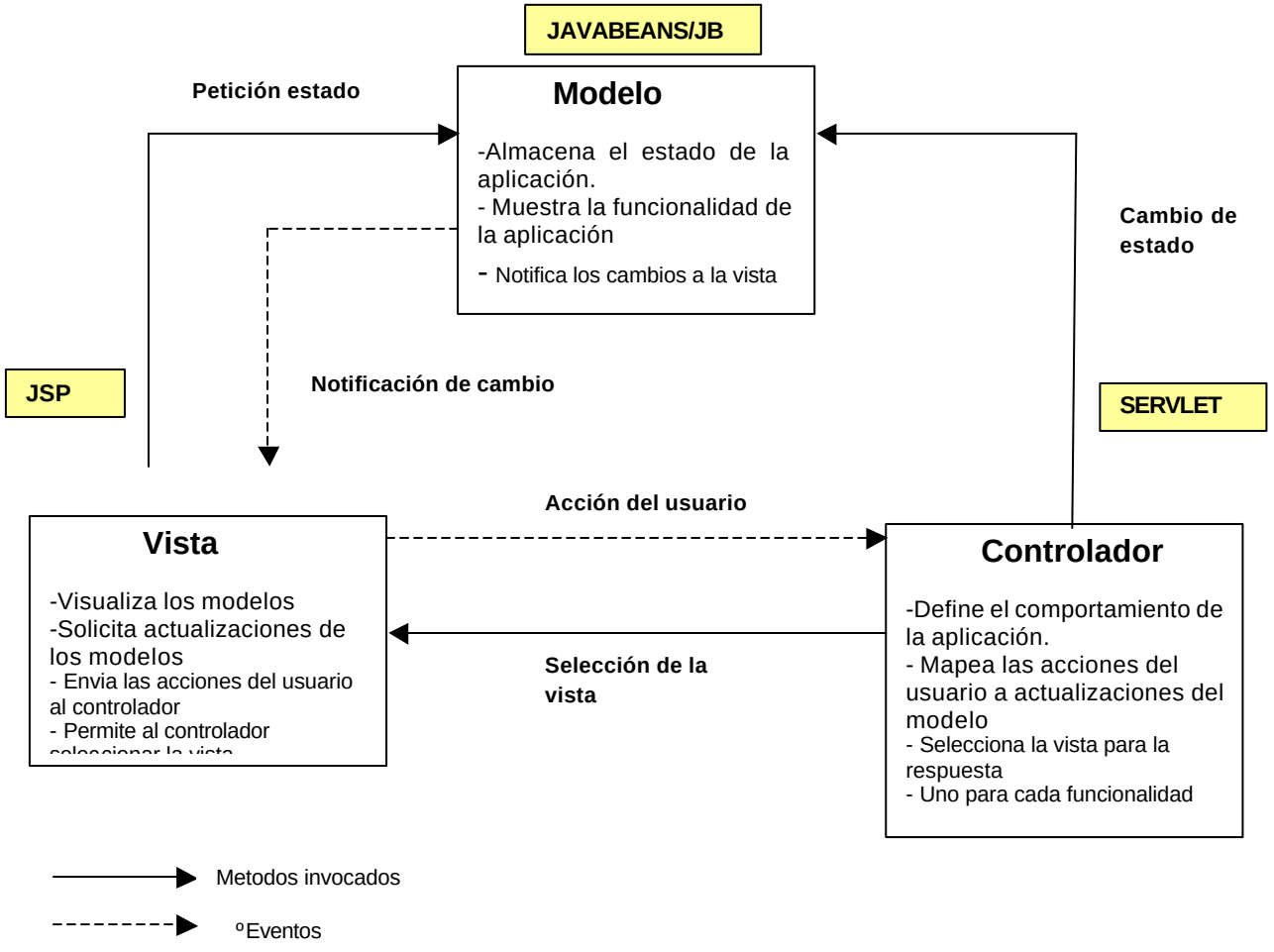
### 2.3.1.- Introducción a Struts

***¿Qué es Struts?***

Es una extensión de clases que implementan el patrón de la arquitectura Modelo-Vista-Controlador (MVC) en Java que surgió para mejorar los problemas del control de flujo que existían en las aplicaciones webs.

***Recordatorio del modelo-vista-controlador***

Es un patrón que define como independiente el Modelo (Objetos de Negocio), la Vista (Interfaz de Usuario) y Controlador ( controlador de flujo de trabajo).



En una aplicación Web esto seria así, el navegador genera una solicitud que es atendida por el Controlador (servlet), este se encarga de analizar la petición, hacer lo que corresponda y llamar al Modelo correspondiente pasándole los parámetros enviados. El Modelo utiliza los objetos de negocio para concretar la tarea y le devuelve un resultado al Controlador quien decidirá la Vista a mostrar.

***¿Qué nos proporciona Struts?***

- Un controlador (servlet) ya implementado del cual podemos heredar para modificar lo que sea necesario.

- El flujo de trabajo (workflow) se puede programar desde un fichero XML. En el fichero de configuración viene entre otras cosas los mapeos (org.apache.struts.action.ActionMapping) correspondientes a cada tipo de petición. Cuando al servlet controlador de Struts le llega una petición este analiza el fichero de configuración (struts-config.XML) y mira que es lo que tiene que hacer y enruta la solicitud HTTP a otros objetos o JSP o a subclases de org.apache.struts.action.Action.

- La Vista se hace con una serie de tags que están definidos por struts con el objetivo de no usar tantos scriptlets (<% %>) lo cual hace que la aplicación sea más mantenible y mejore el rendimiento.

- Define un conjunto de clases bean formulario (org.apche.struts.action.ActionForm) que almacenan fácilmente los datos de un formulario de entrada. Este bean es almacenado para que pueda ser usado en todo momento, se guarda en la sesión, por ejemplo. Este bean puede ser usado por un JSP para mostrar valores, por un Action para validar datos.

- Proporciona un mecanismo para lanzar y mostrar mensajes de error. Cuando una petición llama un Action que usa un bean de formulario, el servlet recupera o crea el bean y lo pasa al objeto Action. Este chequea el contenido del formulario y si ve que tiene algún error lo almacena en una cola de mensajes que después serán mostrados en el formulario. Una vez realizadas las validaciones correspondientes se reenvía al JSP correspondiente.

- Incluye etiquetas personalizadas que pueden rellenar automáticamente campos de un formulario o un bean de formulario. Para esto el JSP tiene que conocer los nombres de las propiedades.

- Struts nos ofrece una manera de manejar los formularios multipart (cuando queremos cargar ficheros desde una página web) de una forma fácil y simple.

- Las etiquetas de struts están hechas de forma que se pueda usar las características de internacionalización incluidas en la plataforma Java. Es decir, el multilenguaje.

- Struts proporciona un simple almacén (pool) de conexiones JDBC. A través de métodos de ActionServlet como *findDataSource().*

### 2.3.2.- Introducción al modelo

Podríamos definir el modelo como un conjunto de objetos, conceptos o visto de forma simple un conjunto de cosas y las actividades que se pueden realizar para cambiar esas cosas.

Este conjunto de cosas en un determinado momento, con unos determinados valores, podríamos decir que son el estado interno de un sistema. Normalmente este estado interno se representa con JavaBeans, con propiedades que representan detalles del estado.

En aplicaciones simples las acciones disponibles podrían ser realizadas por métodos dentro de la clases Action. Pero esto sólo está bien cuando la lógica es muy simple. Pero es mucho mejor separar la lógica de negocio de los objetos Actions.

### 2.3.3.- Introducción a la Vista

En una aplicación basada en Struts la vista esta compuesta por JSPs.

Struts incluye una extensa biblioteca de etiquetas personalizadas que nos facilita la implementación de la interfaz de usuario multiidioma.

### 2.3.4.- Introducción al Controlador

Es el componente principal de los Struts, es un servlet de la clase ActionServlet. El servlet esta configurado definiendo un conjunto de objetos ActionMapping, estos ActionMapping están definidos en el fichero de configuración de struts y contienen un path que comparan con la URL de la solicitud entrante y especifica que Action tiene ejecutarse para la URI elegida. Estas acciones (Action) realizan la lógica de negocio, interpretan la salida y reenvían a la Vista correspondiente.

### 2.3.5.- Construcción del Modelo

En general, el modelo es desarrollado usando JavaBeans que soporten todos los requerimientos del sistema.

En los beans un concepto importante es el del ámbito (scope). El ámbito de un beans se puede definir como las reglas que deciden el tiempo de vida y la visibilidad del bean. Esto se puede definir en los JSP usando términos como:

*page* (los beans sólo son visibles dentro de una sola página JSP),

*request* (los beans son visibles dentro de una sola página JSP y en cualquier otro JSP incluido en esta página y en el servlet),

*session* (los beans son visibles en todas las páginas JSP y servlet que están en una session de un usuario)

*application*(los beans son visibles por todo los sitios de la aplicación)

Por ejemplo:

Bean miBean = new Bean();

Request.setAtribute(“bean”,mibean);

Esto lo puedo usar en un JSP así:

<JSP:useBean id=”bean” scope=”request” class=”Bean”/>

### 2.3.5.1.- Tipos de Beans

#### (a) Beans ActionForm

En el workflow de Struts se asume que tenemos definido una clase que herede de la clase ActionForm por cada formulario de entrada que tengamos en nuestras páginas JSP.

Estos beans son declarados en el fichero de configuración de los struts para que se conviertan en ActionMapping y el actionServlet haga lo siguiente por nosotros:

- Mira si en la sesión de usuario hay un ejemplar de bean de la clase apropiada ( la que diga el ActionMapping).

- Si no lo encuentra crea uno automáticamente y se añade a la sesión de usuario.

- Por cada parámetro de la solicitud cuyo nombre corresponda con el nombre de una propiedad del bean, se llamará al correspondiente método setXXX().

- El bean ActionForm actualizado será pasado al método *perform()* de la clase Action.

Al codificar un bean ActionForm tenemos que tener claro lo siguiente:

- En la clase ActionForm no es necesario implementar ningún método específico. Normalmente solo tendrá métodos setXXX() y getXXX(), sin nada de lógica de negocio.

- El objeto ActionForm tiene un mecanismo para validar. Si sobrescribimos un método *validate()* y proporcionamos mensajes de error en el recurso de la aplicación, se validará la entrada del formulario automáticamente usando nuestro método. La validación se puede realizar aquí, en el Action o en ambos sitios.

- Al definir una propiedad en el bean, el nombre de la propiedad y el nombre del campo presente en el formulario tienen que ser los mismos. Es decir, si tenemos un campo en el formulario que se llama nombre, tendremos en el ActionForm una propiedad que se llame nombre y unos métodos setNombre() y getNombre().

- Tenemos que usar el método *validate()* para asegurarnos que están presentes todas las propiedades requeridas y que contienen valores razonables, si la validación no sale satisfactoriamente este formulario no pasará al objeto Action.

Un formulario en el sentido de ActionForm no es sólo una página JSP, sino que se puede usar un sólo ActionForm para distintas páginas JSP que se refieran al mismo formulario. Struts aconseja que para este tipo de formularios en HTML que tengan varias páginas es mejor definir un sólo ActionForm que tenga las propiedades de todos los campos, sin que importe en que página está ese campo. Al igual que sólo debería tener un ActionForm, también sólo debería tener una clase Action a la que se reenvían todas las páginas.

**(b) Beans de Estado del Sistema**

El estado del sistema esta compuesto por uno o varios beans que definen o representan el estado actual. Ejemplo, en un sistema de matriculación por internet, el sistema tendrá que guardar quién es el usuario que se está matriculando, de cuáles asignaturas se está matriculando , en qué horario etc.

Si el sistema no es muy grande con un conjunto de beans podríamos almacenarlo todo.

Otro caso que suele suceder es que la información necesaria este almacenada un una base de datos externa y los beans de estado sean creados y borrados de la memoria del servidor cuando sea necesario.

**(c) Beans de Lógica de Negocio**

Toda la lógica de negocio debería estar encapsulada en JavaBeans diseñados para esto.

Para aprovechar la reutilización de código los beans de negocio deberían estar diseñados e implementados para que no sepan que están siendo ejecutados en un entorno web, es decir no tienen que saber nada de clases como servlet, httpServletRequest, etc. Toda la información que el bean necesite de estas clases debe de ser proporcionadas por nuestra clase Action.

Dependiendo de la complejidad y del ámbito de nuestra aplicación, los beans puede ser los JavaBeans o JavaBeans Enterprise (EJB).

### 2.3.6.- Construcción de la Vista

La vista en los struts esta realizada con JSP. La particularidad que tiene los struts es que proporciona soporte para conseguir aplicaciones internacionalizadas (multi-idioma) .

Struts proporciona facilidades para construir aplicaciones internacionalizadas y localizadas. Ahora explicaremos algunos conceptos importantes que debemos saber:

- *Locale*: Es una clase de java que soporta la internacionalización. Tiene un conjunto de formateo para números y fechas, dependiendo del idioma del locale.
- *ResourceBundle*: Es un herramienta para mensajes en varios idiomas. Existen varios formatos para definir estos mensajes, uno con un fichero de propiedades del tipo nombre=valor y otro con una lista.
- *MessageFormat:* Clase *java.text.MenssageFormat* formatea un mensaje con argumentos especificados en tiempo de ejecución. Es útil porque las frases en distintos idiomas pueden tener las palabras situadas en distinto orden.
- *MessageResources:* Es una clase de Struts, *org.apache.struts.util.MessageResource* nos permite crear un conjunto de paquetes de recursos como una base de datos y nos permite solicitar un string de mensaje particular para un Locale particular, asociado al usuario o al Locale por defecto.

Para crear una aplicación internacionalizada tendremos que tener unos ficheros de recursos dentro de un paquete de recursos. Los ficheros son *NombreDePaqueteDeRecurso.properties* que contiene mensajes del idioma por defecto de nuestro servidor (mensaje.saludo=Hola) y *NombreDePaqueteDeRecurso_XX.properties*, contiene los mensajes referentes a los distintos idiomas. Tendremos un fichero por cada idioma que queramos, sonde XX será el código del idioma ISO (mensaje.saludo=Hello, para inglés).

En el fichero de web.XML tendremos que dejar constancia de la existencia de este paquete con recursos.

```
<servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>
                org.apache.struts.action.ActionServlet
        </servlet-class>

        <init-param>
                <param-name>application</param-name>
                <param-value>
                        dirección relativa del paquete de recursos
                </param-value>
        </init-param>
</servlet>
```

Para facilitar la realización de formularios y la forma de tratar los errores de éstos, struts proporciona unas librerías de etiquetas personalizadas de JSP. Por ejemplo lo que antes en HTML tendríamos que poner así:

```
<input type="text" name="nombre" value="<% Bean.getNombre() %>"/>
```

Ahora con estas librerías pondríamos así:

```
<HTML:text property="nombre"/>
```

Mucho más fácil.

Si queremos usar estas librerías que nos facilitan el trabajo deberemos incluir en el JSP la directiva *taglib* que le dice al compilador de JSP donde encontrar la librería de tag. Struts dispone de varias librerías de las cuales las más usadas son *struts-HTML, struts-logic, struts-bean.*

Algunas etiquetas de *struts-HTML*:

- HTML:errors : Esta etiqueta muestra los mensajes de error guardados por un componente de la lógica de negocio o no muestra nada si no se ha generado ningún error.
- HTML:form : Asocia todos los campos que hay dentro del formulario a un FormBean con ámbito de sesión.
- HTML:text : Es igual que el input type=”text” el valor será recogido por el FormBean correspondiente.
- HTML:file : Es como la etiqueta input type=”file” de HTML, nos sirve para cargar ficheros. Struts lo hace de forma simple utilizando la clase *FormFile*. En el FormBean correspondiente al campo file la propiedad tiene que ser de tipo FormFile. Struts no da métodos para manejar el fichero correspondiente.
- HTML:link : Genera una etiqueta <a> de HTML y automáticamente aplica la sobreescritura de urls si el navegador no acepta cookies.

Algunas etiquetas de *struts-logic:*

- l**ogic:iterate :** Repite el cuerpo de la etiqueta una vez por cada elemento de una colección especificada (Enumeration, HashTable, Vector, Array).
- **Logic:present:** Dependiendo de un atributo que tiene se muestra el cuerpo de esta etiqueta.
- **Logic:nopresent :** Contrario al present, muestra el contenido de la etiqueta si el atributo no existe.

Los formularios se validan automáticamente gracias al método *validate()* del FormBean asociado al formulario. Este método es llamado por el controlador después de que se hayan rellenado las propiedades del bean pero antes de llamar al execute*()* del Action. Si este método encuentra errores el método execute() no es ejecutado y se muestra al usuario los errores cometidos en la etiqueta HTML:errors.

### 2.3.6.1.- Construcción del Controlador

Struts incluye un servlet que implementa la función principal de mapeo de una solicitud URI a una clase Action.

Para que todo funcione nosotros tenemos varias tareas a realizar:

- Escribir una clase Action por cada solicitud lógica que pudiera ser recibida
- Configurar un ActionMapping por cada solicitud lógica que pudiera ser enviada. Esto se hace en el struts-config.XML
- Actualizar el web.XML para que la aplicación incluya lo que necesite de struts.

### 2.3.6.2.- La clase Action

El método importante de esta clase es *public ActionForward perform(ActionMapping mapping, ActionForm form, HttpServletRequest req, HttpServletResponse res) throws IOException , ServletException;*

En el método *execute()* lo que se debería hacer es lo siguiente:

- Validar la sesión del usuario, es decir que no falta ningún valor necesario.

- Se termina de validar las propiedades del bean formulario. Si hay algún problema se almacenan los mensajes de error y se reenvía la petición al formulario para que se puedan corregir los errores.
- Realizar el procesamiento de los datos que queramos según la lógica de negocio.
- Actualizar los objetos que sean necesarios.
- Devolver un objeto ActionForward apropiado que identifica la página JSP a la que se devuelve el control. Esto lo podremos obtener llamando al *findForward()* o al ActionMapping.

Es importante que las clases Action no sea demasiado largas por lo que es mejor no meter dentro de estas clases toda la lógica de negocio, es mucho mas limpio hacer que sea independiente de la lógica de negocio.

### 2.3.7.- Configuraciones necesarias

Antes de realizar una aplicación con Struts es necesario realizar varias configuraciones. Estas configuraciones incluyen tanto configuraciones de los struts como la del archivos de descriptor de despliegue de la aplicación web.

#### 2.3.7.1.- Web.XML

Tenemos que definir el ActionServlet que es el que vamos usar.

```
<servlet>
   <servlet-name>action</servlet-name>
   <servlet-class>
         org.apache.struts.action.ActionServlet
    </servlet-class>
    <init-param>
         .
         .
         .
     </init-param>
   </servlet>
```

Los parámetros de inicialización *init-param* soportados por ActionServlet son los siguientes:

- *application*: El nombre de la clase base del paquete de recursos para la internacionalización. Por defecto no coge ningún valor.

- *bufferSize*: El tamaño del buffer de entrada usado para procesar upload de archivos. Por defecto es 4096.

- *config*: Path relativo al contexto del recurso que tiene la configuración de los struts. Por defecto es /WEB-INF/struts-config.XML

- *content*: Tipo de contenido por defecto y codificación de la respuesta. Por defecto text/HTML.

- *debug*: Nivel de detalle para la depuración del servlet. Por defecto 0.

- *detail* : Nivel de detalle para la depuración para el Digester ( quién realiza los mapping). Por defecto 0.

- *factory*: Nombre de la clase Java del MessageResourcesFactory usado para crear el objeto MessageResources de la aplicación. Por defecto org.apache.struts.util.PropertyMessageResourcesFactory.

- *formBean*: El nombre de la clase java de la implementación de ActionFormBean a utilizar.

- *forward*: El nombre de la clase java de la implementación de ActionForward

- *locale* : Si se selecciona a true y hay una sesión de usuario almacena el objeto Locale apropiado bajo la clave Action.LOCALE_KEY. Por defecto true.

- *mapping* : El nombre de la clase java de la implementación de ActionMapping.

- *maxfileSize* : El tamaño para los archivos aceptados para upload. Por defecto 250M

- *nocache*: Si se selecciona a true se evita que se guarde la información en la caché.

- *Validate*: Para indicar si se esta usando el nuevo formato de archivo de configuración. Por defecto true.

- *Validating*: Para indicar si se debe usar un parser XML para procesar archivos de configuración.

Esto debe ser puesto para que cuando encuentre una uri que termine en .do pase a través del servlet action.

```
<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

También tenemos que añadir en el web.XML las sentencias necesarias para que que encuentre las taglibraries.

```
<taglib>
  <taglib-uri>/tags/struts-bean</taglib-uri>
  <taglib-location>/WEB-INF/struts-bean.tld</taglib-location>
</taglib>
```

```
<taglib>
  <taglib-uri>/tags/struts-HTML</taglib-uri>
  <taglib-location>/WEB-INF/struts-HTML.tld</taglib-location>
</taglib>

<taglib>
  <taglib-uri>/tags/struts-logic</taglib-uri>
  <taglib-location>/WEB-INF/struts-logic.tld</taglib-location>
</taglib>

<taglib>
  <taglib-uri>/tags/struts-nested</taglib-uri>
  <taglib-location>/WEB-INF/struts-nested.tld</taglib-location>
</taglib>

<taglib>
      <taglib-uri>/tags/struts-tiles</taglib-uri>
    <taglib-location>/WEB-INF/struts-tiles.tld</taglib-location>
</taglib>
```

### 2.3.7.2.- Struts-config.XML

Struts incluye un módulo Digester que es capaz de leer el fichero struts-config.XML de los mapeos deseados creando los objetos deseados como los ActionMapping.

```
<?XML version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.0//EN" "http://jakarta.apache.org/struts/dtds/struts-config_1_0.dtd">
```

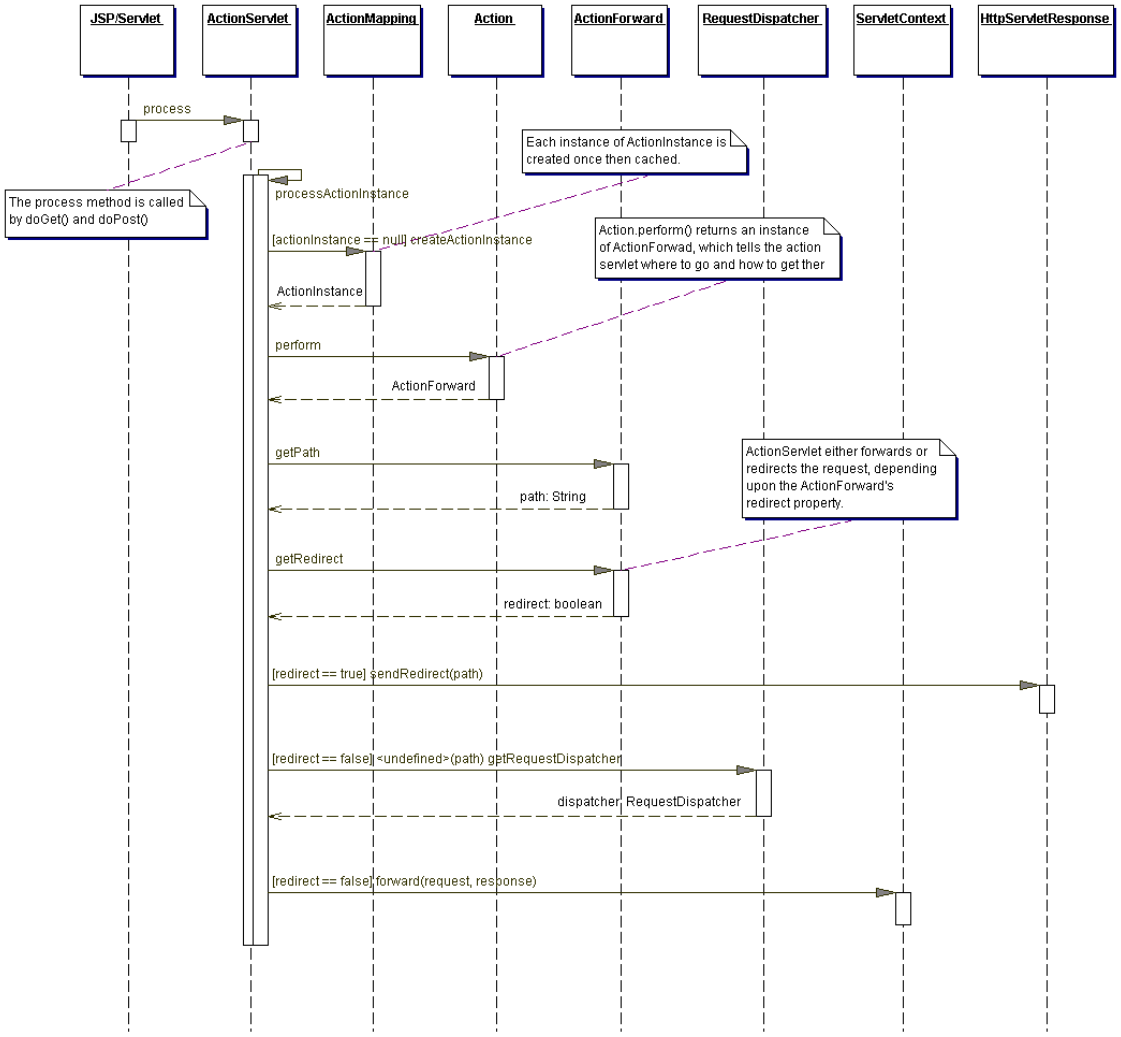
```
<struts-config>
  <form-beans>
        <form-bean name="loginForm" type="src.LoginForm"/>
  </form-beans>

  <global-forwards>
        <forward name="logon" path="/Login.struts"/>
  </global-forwards>

  <action-mappings>
              <action path="/Login" type="src.ActionLogin"
        name="loginForm" scope="request"
              validate="true"
              input="/JSP/Login.JSP"/>

  </action-mappings>

</struts-config>
```

- <form-beans> : se usa por cada bean de formulario que se use en la aplicación. El atributo name es el nombre lógico del bean y el atributo type es el nombre completo de la clase.

- <global-forward> : se usa para crear mapeos de nombres lógicos para páginas JSP usadas comúnmente.

- <action-mapping> : entre estas etiquetas se definen las clases Action, se usa un elemento <action> por cada clase Action que tengamos. En el atributo path se pone el path de la clase Action en relación al contexto de la aplicación, el atributo type define el nombre totalmente cualificado de la clase action y el atributo name define el nombre del elemento formbean correspondiente a esta acción.

## 2.3.8.- Diagrama de secuencia de un ciclo en el MVC de struts.



# Sección IV. Especificación del sistema.

# 1.- Actores

En nuestra ampliación del proyecto <E-Aula> entran en juego los mismos actores que en la implementación original.

## 1.1.- Alumnos :

Constituyen el grupo más amplio de usuarios del sistema. El fin último de esta aplicación es que estos usuarios adquieran los conocimientos presentados en el sistema de la forma que más se adapte a sus preferencias. Por ello, en este proyecto se ha hecho especial énfasis en las características de accesibilidad, las cuales podrá definir el alumno.

## 1.2.- Tutores :

Los tutores son representados por los docentes que supervisan la evolución del alumno durante el curso. En este proyecto se presentan dos funciones que les serán muy útiles a éstos. Una es la elaboración de estadísticas con datos relativos a los alumnos que facilitarán en gran medida el seguimiento que deben realizar todos los tutores sobre sus alumnos. La otra es la sección de envío de correos electrónicos desde la cual un tutor podrá ponersé en contacto con sus alumnos.

## 1.3.- Administradores :

Los administradores del sistema dispodrán de tres funcionalidales muy importantes para la integración de los alumnos de esta aplicación en otros sistemas remotos de e-learning y viceversa, pues podrán exportar la información de un alumno al estándar LIP, asi como importarla desde el estándar al sistema. Teniendo además la posibilidad de eliminar alumnos dentro del sistema.

# 2.- Casos de uso

## 2.1.- Importación del perfil de un alumno

**Actores** : Administrador.

**Desarrollo**: El actor entra en el sistema y accede a la sección de importación de información de alumnos, donde deberá enviar un documento en formato LIP por medio de un formulario. El sistema valida el documento y si es correcto se le notifica al actor la correcta importación de la información, pudiendo ocurrir dos cosas:

- Si el alumno ya existe en el sistema se añade la información académica nueva.

- Si es un alumno nuevo se le da de alta en el sistema introduciendo únicamente la información académica relativa a los cursos impartidos en el sistema actual.

## 2.2.- Exportación del perfil de un alumno

**Actores** : Administrador.

**Desarrollo**: El actor entra en el sistema y accede a la sección de exportación de alumnos donde se le muestra una lista de los alumnos dados de alta en el sistema, así como la opción de poder visualizar todos los ficheros de los alumnos exportados hasta el momento.

El A*dministrador* solicita exportar la información de un alumno en particular o ver todos los ficheros. Si escoge exportar el sistema creará un archivo en formato LIP con toda la información del alumno, tanto de identificación como académica. Si eligiera ver todos los ficheros se le mostraría un listado con todos estos ficheros.

## 2.3.- Eliminación de un alumno

**Actores** : Administrador.

**Desarrollo**: El actor entra en el sistema y accede a la sección de eliminación de alumnos donde se le muestra un listado de los alumnos dados de alta en el sistema. El A*dministrador* solicita eliminar un alumno en particular. El sistema dará de baja el alumno y creará una copia de seguridad de sus datos en formato LIP.

## 2.4.- Visualización de estadísticas

**Actores** : Tutor.

**Desarrollo**: El actor entra en el sistema y accede a la sección de estadísticas donde se le muestra un listado de los cursos. El *Administrador* solicita un curso en particular. A continuación podrá elegir una de las siguientes opciones:

- Numero de conexiones de los alumnos.
- Tiempo dedicado por cada uno de los alumnos.
- Contribuciones de los alumnos a los foros.
- Ítems más visitados.
- Organizaciones más visitadas.
- Porcentaje de suspensos.
- Número de conexiones por tipo de dispositivo.

Para el desarrollo de esta caso de uso se deberán mostrar gráficos que resuman los datos (en los caso en los que sea posible).

## 2.5.- Envío de email a los usuarios de un curso

**Actores** : Tutor.

**Desarrollo**: El actor entra en el sistema y accede a la sección de envío de mail donde se le muestra un listado de los cursos. El *Administrador* solicita un curso en particular. El sistema presenta un listado de todos los alumnos matriculados en ese curso.

El tutor podrá seleccionar de ese listado a los alumnos a los que desee enviar el mail. También dispondrá de opciones de filtrado para enviar el mail sólo a los alumnos activos, a los alumnos activos desde una cierta fecha o a los alumnos matriculados desde una fecha concreta.

Una vez seleccionados los destinatarios, el tutor deberá cumplimentar el asunto y el texto correspondiente al mail y pulsar sobre el botón de envío.

## 2.6.- Definición de perfiles

**Actores** : Alumno.

**Desarrollo**: El actor entra en el sistema y accede a la sección de definición de perfiles donde se mostrará una lista con sus perfiles pudiendo añadir perfiles, modificarlos o borrarlos.

Estos perfiles son las preferencias a la hora de presentarse los contenidos, que tienen los alumnos. Cuando un alumno se conecte a un curso bajo un cierto perfil, el sistema reconocerá el perfil seleccionado y cambiará de apariencia.

# 3.- Otras funcionalidades

Además de los casos de uso citados anteriormente se implementan otras funcionalidades en el sistema.

## 3.1.- Interfaz de eventos.

Se debe implementar un interfaz que permita el manejo de algunas de las tablas del sistema.

Si en un momento dado se produce un evento en el sistema se debe poder reflejar en la base de datos los efectos del mismo.

## 3.2.- Detección de dispositivo.

Es necesario el desarrollo de un módulo capaz de reconocer el dispositivo con el que se está conectando el usuario. Esta información se utilizará en dos sentidos.

- Para estadísticas. Se utilizará para poder mostrar datos de conexiones. Un caso de uso de los citados anteriormente (Estadísticas del número de conexiones por tipo de dispositivo) requiere dicha información.

- Adaptación de la presentación de los contenidos. En base al tipo de dispositivo con el que se está conectando el usuario se presentará la información de una manera u otra.

## 3.3.- Transformación de la representación.

La información mostrada al usuario variará en función de las preferencias del mismo y del dispositivo de conexión.

# Sección V. Diseño e implementación.

# 1.- Módulo de importación y exportación de archivos LIP.

El objetivo consiste en diseñar un módulo capaz de importar alumnos desde ficheros XML, con formato LIP, a la base de datos del sistema y viceversa.

Este apartado tiene dos casos de uso claramente diferenciados:

- Importación de ficheros XML-LIP.
- Exportación a ficheros XML-LIP.

## 1.1.- Importación desde ficheros con formato LIP.

Sólo los administradores tendrán acceso a esta opción.
Una descripción informal del proceso podría ser la siguiente:

- El actor informa a la aplicación de la localización del fichero XML.
- Debemos leer el fichero, comprobando que la estructura sea válida.
- Una vez comprobado que el XML sigue la estructura indicada por el estándar LIP, extraemos la información del alumno.
- Si el alumno ya existía en la base de datos, añadimos la información académica nueva que pudiera tener e informamos de ello al actor. Si no existiera debemos agregar un nuevo alumno al sistema.

- Almacenamos dicha información en la base de datos. Debemos tener en cuenta que solo se almacenará la información académica relativa a cursos que existan en el sistema.

- Acabamos de añadir un nuevo usuario en el sistema, con lo que debemos asignarle un nombre de usuario y una contraseña. La contraseña la generamos aleatoriamente, almacenando en la BBDD el código MD5 de la misma. Para generar el nombre de usuario seguimos el siguiente algoritmo: probamos con el nombre del alumno, si ya existe añadimos un número detrás (comenzando con el 1), y probamos hasta que el nombre de usuario no exista en el sistema (por ejemplo, Juan Martín Churrua → juan, o juan1, o juan2., etc).

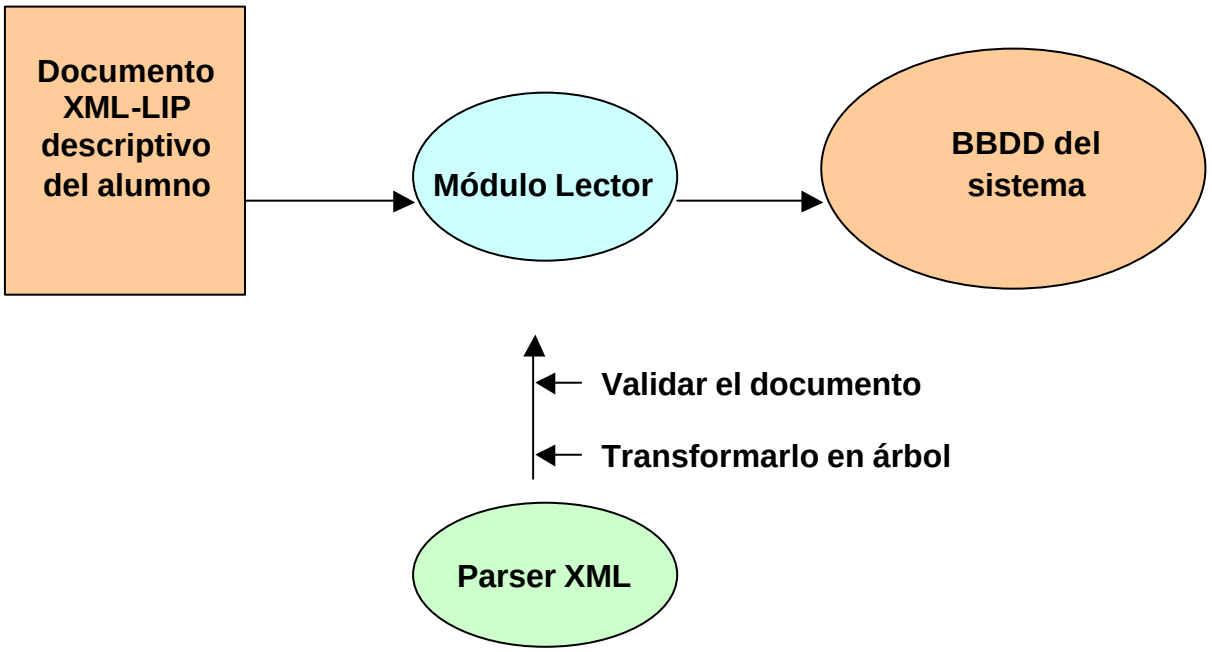
- Se muestra al actor el nombre completo, nombre de usuario y la contraseña del nuevo alumno.

### 1.1.1.- Información almacenada en XML.

Para implementar este módulo hemos utilizado la versión 2.5 del paquete Xerces-J.

El action "ActionImporter" utiliza un objeto de la clase eaula.lip.util.LectorLip para leer el documento XML.

La clase LectorLip se apoya en el DOMParser de Xerces para poder "parsear" el documento y extraer la información correspondiente.




El objeto DOMParser convierte un documento XML en un árbol de tipo Document.

Recorremos este árbol en busca de la información que solicitamos, utilizando el API que nos proporciona Document para desplazarnos por el árbol y extraer la información de los elementos y los atributos de los elementos.

Una de las ventajas más importantes que nos ofrece el API Xerces (no obstante éste ha sido el motivo de su elección), es la capacidad de validar un documento XML frente a un esquema o documento XSD. De éste modo, antes de extraer ninguna información del XML, primero comprobamos que sea un documento LIP-ACCLIP válido contrastándolo frente a los XSD.

De este modo no aseguramos de que el árbol generado seguirá una estructura dada de ante mano (la estructura que nos indica el estandar LIP conjunto con el ACCLIP).

### 1.1.2.- Un nuevo usuario en el sistema.

No debemos perder de vista lo que el módulo pretende como objetivo final: “Añadir un nuevo alumno en el sistema“. Los usuarios del sistema deben tener un nombre de usuario y contraseña asignado.

El nombre de usuario y la contraseña no lo extraemos del documento XML (igualmente no lo escribiremos al exportar). Cuando importamos un nuevo alumno primero debemos comprobar si ya existía en el sistema, en cuyo caso no hacemos nada, se informa al actor del nombre de usuario del alumno y ya está.

En caso de que sea un nuevo alumno en el sistema debemos darle un nombre de usuario y contraseña. El nombre de usuario se asigna como se ha descrito anteriormente, y la contraseña se genera aleatoriamente. Se informa al actor de los dos campos y se almacena en la BBDD el nombre de usuario y la codificación MD5 de la contraseña.

## 1.2.- Exportación de un alumno a formato LIP.

La finalidad de este módulo es realizar la exportación de la información contenida en la base de datos del sistema acerca de un alumno a formato LIP.

El administrador accedería al sistema y podría realizar dos acciones :

- Seleccionar el login del alumno cuya información desee exportar a formato LIP.

Al seleccionar el alumno se ”despierta” al action “ActionExporter”. Lo primero que hace este método es fijar el directorio en el cual se va a exportar el fichero LIP. Para ello comprueba la ubicación donde está instalada la aplicación y , si es la primera vez que exportamos un fichero, creará dentro de esa

ubicación un directorio denominado “ficherosLip”. Si este directorio ya existiera, simplemente lo fijaría como ubicación de destino para el fichero LIP.

A continuación comprobamos si se ha seleccionado un alumno, en caso negativo se mostraría un JSP informando del error. Si el alumno se ha seleccionado correctamente el siguiente paso es establecer la ruta donde se encuentran los esquemas que servirán para validar si el fichero LIP que vamos a generar sigue las especificaciones establecidas.

Una vez fijados estos parámetros la clase “ActionExporter” se apoya en la clase “eaula.lip.util.Escritor” para realizar la creación del fichero en formato LIP. Esta clase “Escritor” realiza todas las consultas pertinentes sobre la base de datos relativas al alumno que le ha fijado “ActionExporter”. De este modo consigue extraer toda la información tanto de identificación, como académica, como de accesibilidad, concernientes al alumno. Seguidamente comienza a generar el fichero LIP siguiendo las especificaciones del IMS e incluyendo en el mismo la ruta donde están ubicados los esquemas necesarios para validar el documento. Cuando termina de crear el fichero LIP lo ubica en el directorio marcado por la clase “ActionExporter” finalizando así el proceso de exportación.

El administrador recibirá un mensaje informándole de la correcta exportación del fichero, así como un vínculo para poder descargar dicho fichero a una ubicación distinta si así lo desea.

- Seleccionar la opción de ver todos los ficheros LIP exportados hasta el momento .

Al pulsar sobre la opción de visualizar todos los fichero LIP el sistema presenta un nuevo JSP dónde aparece un listado de todos aquellos ficheros LIP que han sido exportados hasta el momento. Estos ficheros contendrán información relativa únicamente a los alumnos activos en el sistema.

El listado de los ficheros serán en realidad vínculos a dichos ficheros para que el administrador pueda descargárselos a la ubicación que considere adecuada.

## 2.- Módulo de eliminación de un alumno del sistema.

El modulo de eliminación es el encargado de dar de baja un alumno en el sistema, asi como de generar una copia de seguridad en formato LIP de toda la información que tengamos en la base de datos sobre dicho alumno.

El proceso sería el siguiente :

- El administrador accede a la sección de eliminación.
- Seguidamente se le presenta un JSP con un listado de todos los usuarios activos en el sistema.
- El adminsitrador selecciona uno de los alumnos. Si por error no escogiera ningún alumno se le informaría mediante un mensaje.

Al seleccionar el alumno se activa el método execute de la clase ”ActionDelete”. Dicho método realiza prácticamente las mismas operaciones que el ”ActionExporter”, cambiando el directorio de destino en el que se ubicará la copia de seguridad de la información del alumno en formato LIP, que en este caso sería ”ficherosLipEliminados”. Además de realizar la exportación de los datos del alumno, se encarga también de colocar el bit “Eliminado” a 1 dentro de la fila correspondiente al alumno en la base de datos. Por otro lado este método también se encarga de comprobar si había en el directorio “ficherosLip” alguno que perteneciese al alumno que se desea eliminar, en cuyo caso eliminaría también dicho fichero.

## 3.- Módulo de estadísticas

El objetivo de este módulo es mostrar datos estadísticos sobre distintos conceptos del sistema <e-Aula> que pueden ser útiles para los profesores.

Tendremos las siguientes opciones dentro de este módulo:

- Número de conexiones de un alumno al curso

- Número de contribuciones al foro del curso

- Organizaciones más visitadas del curso

- Porcentaje de suspensos de los exámenes realizados en el curso

- Tiempo dedicado por el alumno al curso

- Items más visitados del curso

- Notas de los exámenes del curso

- Número de conexiones por tipo de dispositivo

Basándonos en el marco de trabajo que nos ofrece Struts, hemos optado por tener para cada una de las distintas opciones un objeto de tipo *Action* que realiza la conexión con la base de datos para extraer la información necesaria para cada opción. Esta información la almacenamos en objetos de tipo *Bean* para la comunicación entre la parte de la lógica de negocio y la parte de presentación de la aplicación.

Para la visualización de los datos se ha optado por el uso de gráficas en aquellos casos en los que se ha considerado útil. Estos casos son los siguientes:

- Número de conexiones por tipo de dispositivo

- Porcentaje de suspensos de los exámenes realizados durante el curso.

- Organizaciones e items más visitados, dando también opción a visualizar cuantas visitas tienen cada uno de las organizaciones y de los items.

Para la realización de estas gráficas hemos usado una librería de libre distribución llamada JFreeChart . Esta librería soporta distintos tipos de gráficas (de barras, de tarta, de línea...) y se pueden usar en aplicaciones, applet, servlet y JSP.

Este módulo facilita la tarea de supervisión de un curso por parte del profesor ya que podrá obtener información muy útil con un simple click.

Para evaluar a los alumnos, el profesor puede tener en consideración distintos factores como las contribuciones en el foro o el tiempo que han dedicado al curso.

A partir de la información sobre el número de conexiones según los dispositivos, el profesor puede optar por mejorar el contenido de los cursos para que se visualicen mejor desde dichos dispositivos. Es decir, si ve un aumento de los accesos vía PDA podría realizar los cursos con menos contenido en cada página y con más paginas para asi facilitar a los alumnos que utilicen ese medio.

## 4.- Módulo de notificaciones por correo electrónico

Este módulo tiene como objetivo el envío de correos a los alumnos matriculados en un curso por parte del profesor.

El profesor puede elegir los destinatarios de los correos a través de unos criterios de búsqueda.

Para ello dispone de una lista con los alumnos matriculados en el curso y unos botones donde se puede filtrar la lista de estos alumnos. Los criterios de filtrado son, sólo enviar correo a los alumnos actualmente activos en el sistema, enviar correos a los alumnos que llevan activos en el sistema desde una cierta fecha o enviar correos a los alumnos que están matriculados desde una cierta fecha.

Hemos usado dos objetos *Action*. El objeto *ActionDatosCorreos* es el encargado de realizar la conexión con la base de datos y obtener la información necesaria tanto para rellenar la lista de alumnos como para realizar el filtrado de los mismos.

El segundo objeto *Action* utilizado es el objeto *ActionEnviarCorreos* que es el encargado de realizar el envío de correos, como su propio nombre indica. Para realizar esta acción hemos usado la librería *JavaMail de java*. Se necesitan los siguientes datos para la configuración del servidor de correo:

- **mail.transport.protocol**: Protocolo de transporte de correo, en nuestro caso es *SMTP (Simple Mail Transport Protocol)*
- **mail.smtp.host** : Dirección del servidor de correo saliente
- **mail.from**: Dirección de correo del remitente
- **mail.user**: Usuario de la cuenta de correo del remitente
- **mail.password**: Contraseña de la cuenta de correo del remitente
- **mail.smtp.port:** Puerto del protocolo smpt, por defecto es el 25
- **mail.debug**: Propiedad booleana para indicar si la librería debe mostrar información de debug o no.

## 5.- Módulo de preferencias

Este es el módulo encargado de añadir, modificar y borrar los perfiles del usuario.

Un usuario puede tener más de un perfil, siendo un perfil un conjunto de características con los valores que el usuario prefiere para estas.

Este conjunto de características tienen una correspondencia directa con alguna de las etiquetas que están definidas por el estándar ACCLIP, explicado anteriormente en el apartado de 1.2.4

Existe un editor de XML ACCLIP diseñado por la universidad de Toronto. Este programa, a través de distintos formularios preguntando por nuestras preferencias construye el documento XML-ACCLIP descriptivo (*http://www.utoronto.ca/atrc/research/Web4All*).

Se tuvo que decidir que características del estándar ACCLIP se usaban en esta versión del editor de preferencias, llegando a la conclusión de usar las siguientes:

- **FontName:** Tipo de letra, teniendo como opción Serif, Sans-serif, Fantasia, Monospace.

- **FontSize:** Tamaño de letra, teniendo como opción Pequeña, Mediana, Grande o Muy grande.

- **BackGroundColor:** Color de fondo de la pantalla, teniendo para elegir una paleta de colores.

- **CursorSize**: Tamaño del cursor, teniendo como opción Pequeño, Grande o Mediano.

- **CursorColor:** Color del cursor, teniendo para elegir una paleta de colores.

- **ForegraundColor:** Color de la letra, teniendo para elegir una paleta de colores.

- **ContentDensity:** Densidad de contenido, teniendo para elegir Mucho, Medio o Poco.

- **ContentView:** Indica si se quieren ver las imágenes o solo el texto, teniendo como opción Imágenes y Texto.

- **StructuralNavigation:** Indica si se quiere tener en la parte superior del curso un índice por el cual poder navegar más fácilmente entre los temas más importantes.

Hemos tenido que introducir una tabla en la base de datos donde almacenamos los valores de estas preferencias para los distintos perfiles. Teniendo un campo para indicar cual es el perfil por defecto al conectarse el usuario.

La tabla de la base de datos tiene un campo para cada una de las preferencias, añadiendo además:

- **User :** Usuario al que pertenece el perfil.

- **NombrePerfil:** Nombre del perfil.

- **PorDefecto:** Perfil por defecto cuando no se ha seleccionado ninguno. Sólo podrá haber un perfil por defecto.

Se tomo la decisión de añadir un perfil en la base de datos que fuera el perfil por defecto si un usuario no tenía ninguno dado de alta o no tenía ninguno por defecto. Este perfil tiene como nombre de usuario *"EAULAtodosEAULA"* y como nombre de perfil *"PorDefectoEAULA"*.

El sistema permite añadir, eliminar y modificar los perfiles de los usuarios. Si un usuario no introduce las características de al menos un perfil se

tomarán los valores por defecto almacenados en el registro citado anteriormente.

# 6.- Módulo reconocedor de dispositivo

Este módulo realiza la función de reconocer el dispositivo desde el que se está realizando la petición al sistema

Para realizar este módulo hemos tenido que realizar en primer lugar una tarea de investigación, ya que el mundo de los dispositivos móviles no es nada trivial y el reconocimiento de éstos es algo en lo que las grandes empresas de telecomunicaciones invierten mucho dinero y recursos.

El reconocimiento del dispositivo mediante el cual se conecta el usuario es necesario para la posterior adaptación del contenido del curso, ya que el contenido variará dependiendo del dispositivo desde el cual se haya conectado.

En la actualidad existen muchos problemas a la hora de identificar un dispositivo sólo a partir de la petición realizada (request). No conocemos ningún estándar que defina como deben ser las cabeceras, los datos que se mandan en la petición, etc.

## 6.1.- Primeras propuestas

En la primera fase del análisis diferenciamos tres maneras de reconocer el dispositivo con sus respectivas ventajas e inconvenientes:

### 6.1.1.- Distinción por user-agent.

Todos los dispositivos al realizar una petición mandan su user-agent, que es el identificador del modelo del terminal. Podríamos tener una tabla en la

base de datos donde se relacionen los user-agent con el tipo de dispositivo que corresponda. Este trabajo sería algo costoso, ya que existen muchísimos móviles distintos y muchas (aunque menos) PDAs diferentes, cada uno con su propio user agent.

### 6.1.2.- Distinción por el sistema operativo

Cada tipo de dispositivo tiene sistemas operativos distintos y nos pueden ayudar a identificar a que tipo pertenece.

El dato del sistema operativo viene en la cadena del user agent. Tendríamos que tener una tabla en la base de datos donde se relacione sistemas operativos con los distintos tipos de dispositivos. Esto sería menos extenso que tener una tabla con los user agent de los dispositivos pero habría que investigar para saber cuales son los sistemas operativos existentes en el mercado actualmente.

### 6.1.3.- Utilización de wurfl

WURFL es un proyecto de software libre consistente en una colección de dispositivos y características (o utilidades) de los mismos

La información es almacenada en un fichero XML (wurlf.XML). Se dispone de un API de acceso con el cual poder hacer consultas. La idea consiste en identificar el dispositivo a través del user-agent, con el cual podemos consultar a wurlf cualquier aspecto técnico necesario para las adaptaciones, tales como ancho de pantalla, lenguaje de marcado, si acepta formato de imagen GIF o no, etc.

Un problema que podemos encontrar es que el fichero XML no es completamente fiable, por ejemplo, no todas las características son correctas.

Para solucionar el problema anterior wurlf dispone de un fichero XML aparte, wurlf_patch.XML, donde podemos redefinir características y añadir nuevos terminales. Sirve para no "manchar" el wurlf.XML pudiendo hacer nuestras propias correcciones.

La utilización de wurfl para nuestros fines conllevaría muchos problemas ya que sólo contiene los dispositivos móviles y habría que añadir los pcs con su user-agent y sus características. Si hiciéramos una petición y el dispositivo no estuviera introducido en el XML no podría reconocerlo y siempre devolvería un dispositivo que existe por defecto (el dispositivo genérico).

### 6.1.4.- Decisión final

Hemos decidido hacer el reconocedor de dispositivo de una forma abierta, para que pueda ser administrable.

Al intentar reconocer un dispositivo buscaremos en la base de datos si está dado de alta el correspondiente user agent que lo identifica, si es así cogeremos los valores concretos de ese dispositivo. Si no está en la base de datos tendremos que pasar al reconocimiento mediante reglas, para intentar ver a qué familia de dispositivos pertenece el que realiza la petición.

Hemos diferenciado cuatro familias de dispositivos basándonos en las características que necesitamos saber sobre estos. Estas familias son las siguientes:

- **PDA**: Esta es la familia de los denominados ordenadores de bolsillo. Hemos querido diferenciarlo ya que tiene un tamaño de pantalla mayor, mayor capacidad de almacenamiento, etc. que el resto de dispositivos móviles.

- **WML**: Esta es la familia de los móviles que utilizan como lenguaje de marcado el WML. Hemos puesto esta familia aparte de la del resto de

móviles porque, aunque sean móviles, tienen muchas particularidades. Sólo entienden el lenguaje de marcado WML, sólo admite imágenes wbmp y además tienen un tamaño de caché muy pequeño. Estos tipos de móviles están en desuso debido a estas limitaciones.

- **MOVIL**: Esta es la familia del resto de dispositivos móviles. Tienen unas capacidades menores que las PDAs y no tienen las limitaciones de los dispositivos WML.

- **PC** : Esta familia será asignada por defecto si antes no se le ha asignado ninguna de las anteriores. Esta familia en nuestro caso es la que más se daría ya que la mayoría de los usuarios se conectarían al curso a través de PCs.

## 6.2.- Modelo de datos

Para realizar el reconocimiento de dispositivos hemos añadido una serie de tablas a la base de datos. A continuación las iremos explicando una a una.

**Tabla DeviceCaract**

Esta tabla contiene la información que necesitamos saber sobre los dispositivos.

Tiene los siguientes campos:

- **idDevice:** Identificador de dispositivo, podrá ser el user agent si se ha dado de alta un dispositivo concreto o el nombre de alguna de las familias, para cuando no está en la base de datos el dispositivo concreto.

- **anchoDisplay:** Ancho de la pantalla.

- **altoDisplay:** Alto de la pantalla.

- **tipoImg :** Identificador del tipo de imagen de la tabla TipoImg.

- **tipoSound:** Identificador del tipo de sonido de la tabla TipoSound.

- **tipoVideo:** Identificador del tipo de video de la tabla TipoVideo.

- **tamCache:** Tamaño de la memoria caché del dispositivo.

**Tabla TipoImg**

Tabla en la que tenemos la correspondencia del identificador del tipo de imagen con la extensión de dicho formato.

Tiene los siguientes campos:

- **tipoImg:** identificador del tipo de imagen del dispositivo
- **format:** Identificador del formato de la imagen de la tabla FormatImg.

**Tabla TipoSound**

Tabla en la que tenemos la correspondencia del identificador del tipo de sonido con la extensión de dicho formato.

Tiene los siguientes campos:

- **tipoSound:** identificador del tipo de sonido del dispositivo
- **format:** Identificador del formato del sonido de la tabla FormatSound.

**Tabla TipoVideo**

Tabla en la que tenemos la correspondencia del identificador del tipo de vídeo con la extensión de dicho formato.

Tiene los siguientes campos:

- **tipoVideo:** identificador del tipo de video del dispositivo
- **format:** Identificador del formato de video de la tabla FormatVideo.

**Tabla FormatImg**

Tabla en la que tenemos la correspondencia del identificador de formato con la extensión de dicho formato.

Tiene los siguientes campos:

- **id:** Identificador del formato.
- **ext:** Extensión o nombre del formato.

**Tabla FormatSound**

Tabla en la que tenemos la correspondencia del identificador de formato con la extensión de dicho formato.

Tiene los siguientes campos:

- **id:** Identificador del formato.
- **ext:** Extensión o nombre del formato.

**Tabla FormatVideo**

Tabla en la que tenemos la correspondencia del identificador de formato con la extensión de dicho formato.

Tiene los siguientes campos:

- **id:** Identificador del formato.
- **ext:** Extensión o nombre del formato.

**Tabla PalPDA**

Tabla con palabras que aparecen muy frecuentemente en los user agent de las PDAs. La usaremos para discriminar cuando es de la familia PDA. Estas palabras tienen que estar pensadas para que sólo aparezcan en dispositivos PDAs.

Tiene los siguientes campos:

- **pal:** Palabra que parece frecuentemente en los user agent de las PDAs.

**Tabla SubUA**

Tabla de correspondencia entre palabra y familia a la que pertenece. Estas palabras deberán aparecer en el user agent de la petición.

Tiene los siguientes campos:

- **subUA:** Palabra que tiene que aparecer en el user agent del dispositivo
- **idDevice:** Familia a la que pertenece si aparece esa palabra.

Las tablas DeviceCaract, TipoImg, TipoSound, TipoVideo, FormatImg, FormatSound, FormatVideo contendrán los datos correspondientes a las familias anteriormente descritas, dejando abierto el sistema para poder

introducir, modificar o eliminar datos de dispositivos concretos si se dispone de la información necesaria.

Las tablas subUA y palPDA contiene datos sacados a través de un estudio sobre los user agent de distintos dispositivos, dejando abierto el sistema para poder introducir, modificar o eliminar los datos.

## 6.3.- Implementación

En este módulo tenemos dos clases importantes que son la clase llamada **DeviceBean** que es el objeto que devolveremos. Este objeto contiene la información correspondiente al dispositivo reconocido. La otra clase que realiza la tarea de reconocer el dispositivo se llama **Reconocedor**.

La clase Reconocedor utiliza un patrón singleton, para asegurarnos que solo existe una instancia de esta clase ya que no son necesarias más y con esto mejoramos la eficiencia.

En el constructor de la clase Reconocedor accedemos a la base de datos para recoger los datos de las tablas subUA y palPDA para así no tenerlos que pedir cada vez que los queramos utilizar sino que al pedir la primera instancia del reconocedor lo almacenaremos en un objeto.

Esta función tiene el siguiente comportamiento:

- Intenta buscar si el dispositivo concreto aparece en la base de datos, es decir si tenemos en la tabla DeviceCaract algún idDevice que coincida con el user agent del dispositivo que realiza la petición. Si es así ya no seguimos buscando y nos quedamos con ese deviceBean.

- Si no ha encontrado el dispositivo concreto vemos si la familia de dispositivo es WML, esto lo sabremos si dentro de sus cabeceras accept aparece **vnd.wap.WML** y no aparece **HTML**. Ya que los

WML no admiten HTML en ninguna de sus versiones. Si ocurre esto, rellenaremos el objeto DeviceBean con los datos correspondientes de la familia WML.

- Si el dispositivo no es de la familia WML, probaremos si es de la familia PDA. Esto lo realizaremos comprobando si en el user agent de la petición aparece alguna de las palabras que están en la tabla palPDA. Si es así, rellenaremos el objeto DeviceBean con los datos correspondientes de la familia PDA.

- Si el dispositivo no es de la familia PDA, probaremos a ver si lo identificamos a través de la tabla subUA. Esto lo haremos comprobando si aparecen algunas de las palabras de esta tabla en el user agent. Si es así rellenaremos el objeto DeviceBEan con los datos correspondientes a la familia que esta asignada a esa palabra.

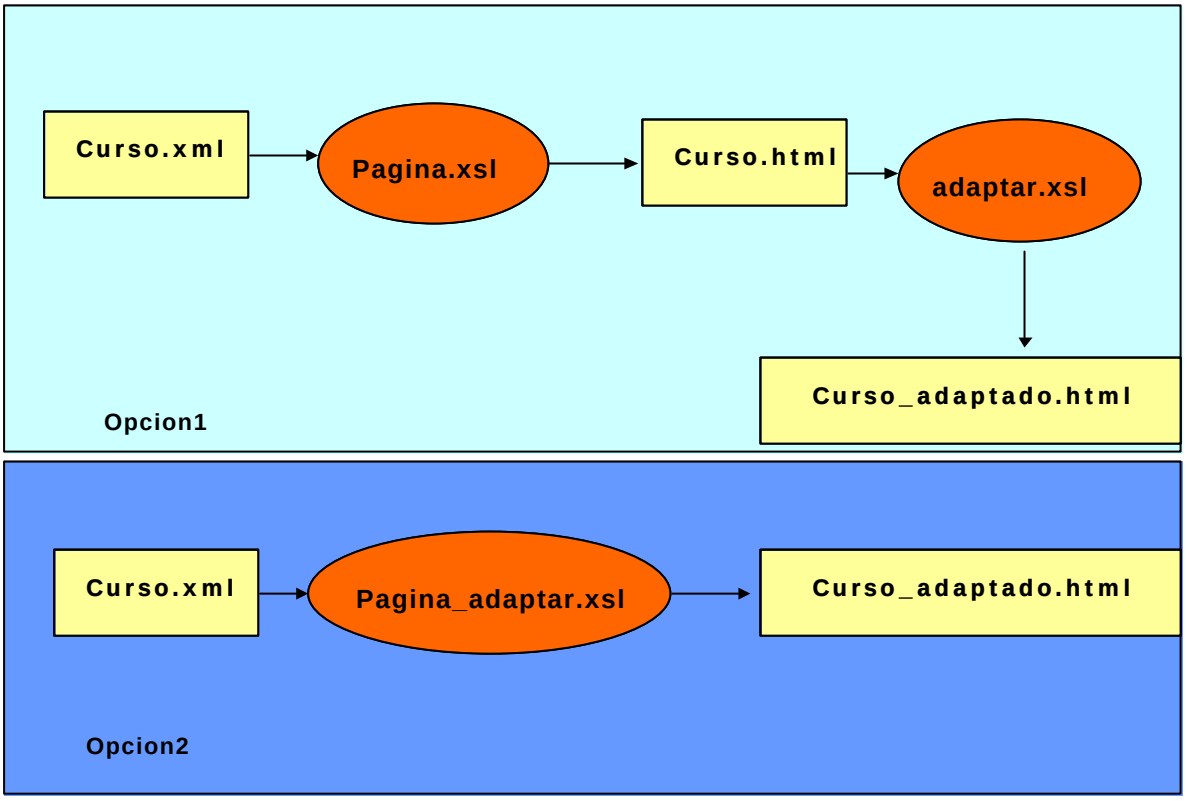
Si no hemos reconocido el dispositivo por ninguna de las anteriores reglas, rellenaremos el objeto DeviceBean con los datos correspondientes a la familia PC.

## 7.- Adaptación según las preferencias del usuario y el dispositivo

El objetivo de este módulo es doble ya que se ha realizado una adaptación en base a las preferencias del usuario, dependiendo del perfil con el cual se haya conectado, y otra en base al dispositivo desde el cual se conecta.

Para realizar esta adaptación se optó por usar transformaciones xstl, ya que los datos estarían en formato XML.

Teníamos dos opciones :




Para implementar la adaptación se eligió la opción 2 , al ser más eficiente, ya que sólo se necesita un paso para la transformación.

Otra decisión de diseño fue la de usar XSL con parámetros. Con ello conseguimos una mayor flexibilidad puesto que se pueden reutilizar las mismas hojas de estilo pasándole distintos parámetros.

Los parámetros son pasados desde la clase *Transformer,* siendo estos parámetros los atributos de las clases *Perfil* y *DeviceBean*.

A continuación explicaremos los distintos parámetros.

**DeviceBean**

- **AnchoDisplay:** Ancho de la pantalla del dispositivo.
- **AltoDisplay:** Alto pantalla del dispositivo.
- **TipoImagenes:** Lista de formatos de imágenes soportados por el dispositivo
- **TipoSonidos:** Lista de formatos de sonidos soportadas por el dispositivo.
- **TipoVideo:** Lista de formatos de vídeo soportados por el dispositivo.
- **TamCache:** Tamaño de la caché del navegador del dispositivo.

**Perfil**

- **FontName:** Tipo de letra
- **FontSize:** Tamaño de letra
- **ForegroundColor:** Color de la letra
- **BackGroundColor:** Color de fondo de la pantalla
- **CursorSize**: Tamaño del cursor
- **CursorColor:** Color del cursor
- **ContentDensity:** Densidad de contenido
- **ContentView:** Indica si se quieren ver las imágenes o solo el texto
- **StructuralNavigation:** Indica si se quiere tener en la parte superior del curso un índice por el cual poder navegar más fácilmente entre los temas más importantes.

Hemos iniciado el camino de la adaptación usando alguno de los parámetros anteriormente descritos, dejando para posteriores ampliaciones del proyecto una base sobre la que trabajar para conseguir realizar la incorporación del resto de parámetros.

A continuación explicaremos los parámetros que hemos aplicado en la adaptación y como los hemos usado:

**TipoImagenes**

En la adaptación, dependiendo del tipo de imágenes que acepte el dispositivo, se ha optado por no mostrar la imagen si el dispositivo no la reconoce, mostrando solamente el título de la imagen. Esto lo vemos a través de las extensiones de las imágenes a mostrar.

**FontName**

Se ha modificado el estilo de la fuente cambiando el atributo font-family por el indicado en el parámetro. Este parámetro no tiene sentido para páginas WML ya que en dichos dispositivos no se puede modificar el tipo de letra.

**FontSize**

Se ha modificado el estilo de la fuente, cambiando el atributo font-size por el indicado en el parámetro. Este parámetro no tiene sentido para páginas WML ya que en dichos dispositivos no se puede modificar el tamaño de letra.

**ForegroundColor**

Se ha modificado el estilo de la fuente cambiando el atributo color por el indicado en el parámetro. Este parámetro no tiene sentido para páginas WML ya que en dichos dispositivos no se puede cambiar el color de la fuente.

**BackGroundColor**

Se ha modificado el color de fondo de la pantalla cambiando el atributo bg-color por el indicado en el parámetro. Este parámetro no tiene sentido para páginas WML ya que dichos dispositivos no se puede cambiar el color de fondo de la pantalla.

**ContentDensity**

Con este parámetro se reduce la cantidad de contenido dependiendo de lo que el usuario desee. Esto se realiza mediante una función de poda para llegar a una profundidad de contenido deseada.

**ContentView**

Si este parámetro llega con la opción de no mostrar imágenes se eliminan las etiquetas < img > y en su lugar se muestra un texto donde se indicará que había una imagen y el título de la misma.

Si el parámetro llega con la opción de mostrar imágenes estas se muestran normalmente.

**StructuralNavigation**

Si este parámetro llega con la opción de mostrar el índice de navegación, se mostrara al principio de la página una lista de enlaces para navegar por los apartados más importantes del curso.

Si el parámetro llega con la opción de no mostrar el índice de navegación no se mostrará la lista de índices.

Dejamos como ideas propuesta para la siguiente fase del proyecto las siguientes:

**AnchoDisplay ,AltoDisplay, TipoImagenes**

Los atributos *AnchoDisplay* y *AltoDisplay* podrían servir para hacer una adaptación de las imágenes al dispositivo. Una idea sería usar una herramienta libre distribución llamada ImageMagick y unas librerías para java llamada JMagick que se comunican con ImageMagick.

Estas herramientas ofrecen la posibilidad de crear, editar y componer imágenes y con ello la posibilidad de cambiarle el tamaño o incluso de formato, si el formato de la imagen no es soportado por dispositivo.

**TamCache**

Este parámetro es muy importante dentro de los dispositivos móviles ya que están bastante limitados por él. Si se quisiera hacer una adaptación teniendo en cuenta este parámetro se debería realizar un proceso donde se calcularía el tamaño de la página incluyendo lo que ocupan las imágenes, videos y todo lo que haya en ellas. Una vez tengamos lo que se suele llamar el "peso" de la página, se debería comprobar que entra en el dispositivo, dependiendo del parámetro TamCache y si no es así, se debería paginar el curso de manera que entre todo pero en varias páginas.

Hemos realizado dos hojas de estilo, una para WML debido a sus especiales características y otra para el resto de dispositivos.

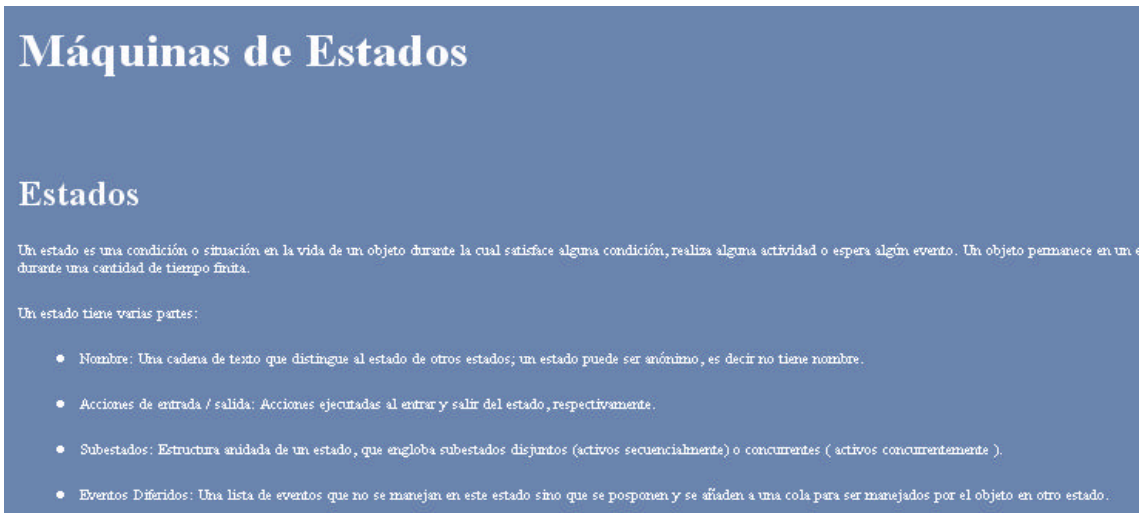
## 7.1.- Ejemplos de adaptacion

**Ejemplo 1**

**FontSize**: Pequeño

**FontName**: serif

**ForeGroundColor**: blanco




**PC**


**MOVIL**


**WML**

**Ejemplo 2**

**FontSize**: Grande

**FontName**: Monospace

**ForeGroundColor**: Amarilla

**BackGroundColor:** Rojo

# Máquinas de Estados

## Estados

Un estado es una condición o situación en la vida de un objeto durante la cual satisface alguna condición, realiza alguna actividad o espera algún evento. Un objeto permanece en un estado durante una cantidad de tiempo finita.

Un estado tiene varias partes:

- Nombre: Una cadena de texto que distingue al estado de otros estados; un est
  puede ser anónimo, es decir no tiene nombre.
- Acciones de entrada / salida: Acciones ejecutadas al entrar y salir del esta
  respectivamente.

**PC**


**MOVIL**


**WML**

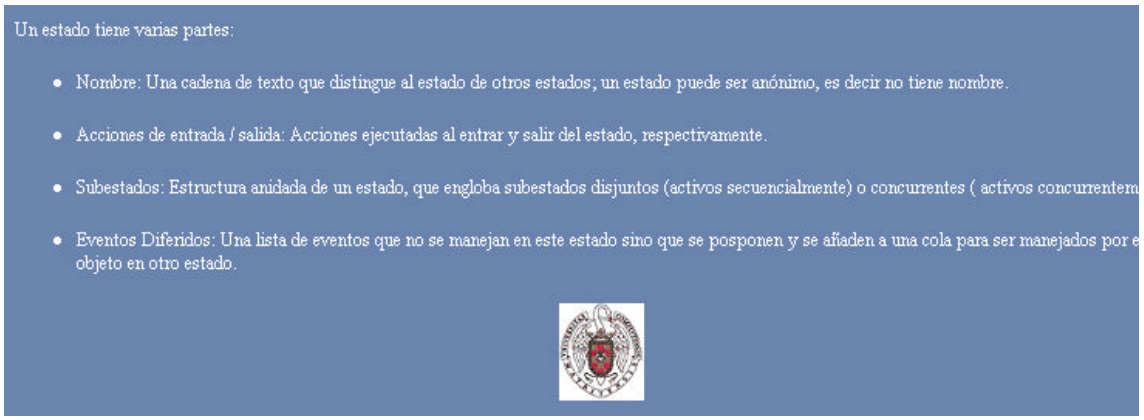
**Ejemplo 3**

**FontSize**: Mediano

**FontName**: serif

**ForeGroundColor**: blanco

**ContentView:** Imagenes




**PC**


**MOVIL**


**WML**

En WML no se muestran las imágenes debido a que no admite formato gif. En el caso de haber puesto contentView=texto saldría igual en MOVIL.

## Ejemplo4

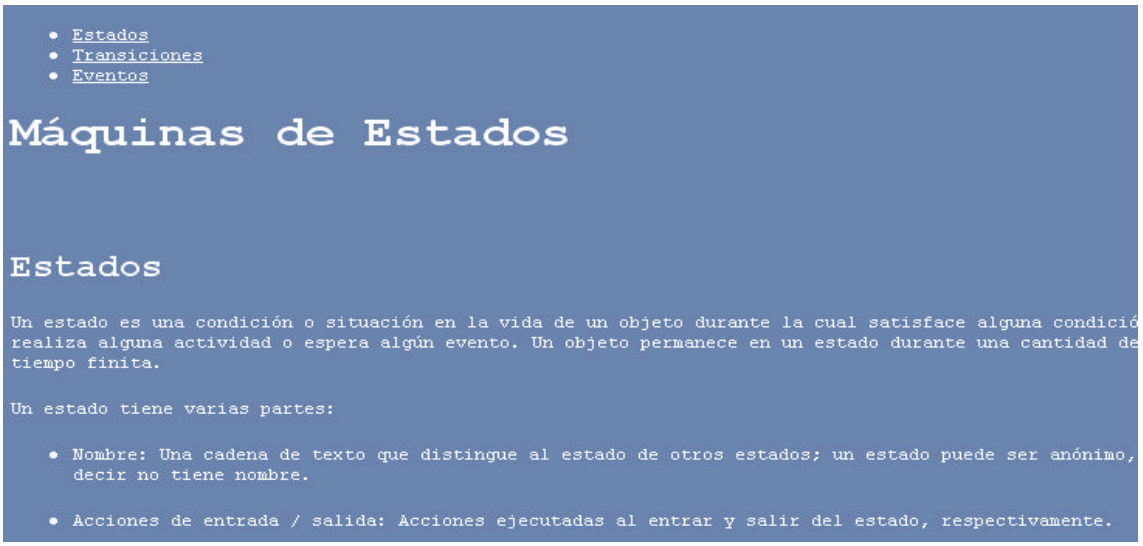
**FontSize**: Mediano

**FontName**: Monospace

**ForeGroundColor**: blanco

**ContentView:** Imagenes

**StructuralNavigation:** Si




**PC**


**MOVIL**

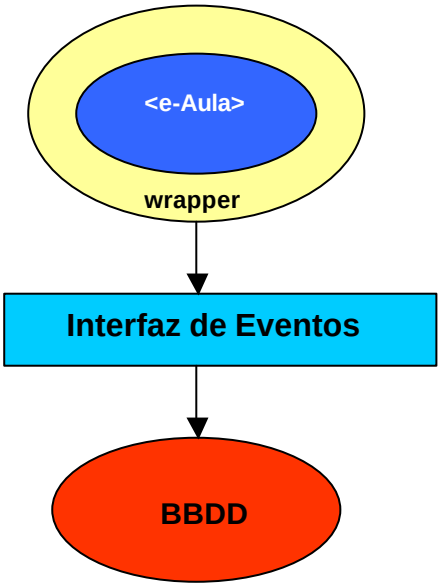

**WML**

## 8.- Interfaz de eventos.

En el desarrollo del proyecto se han generado nuevas tablas en la BBDD y nuevos campos en otras de ellas.

Es posible (y altamente probable) que surja la necesidad de modificar algunos de estos campos desde el núcleo del sistema.

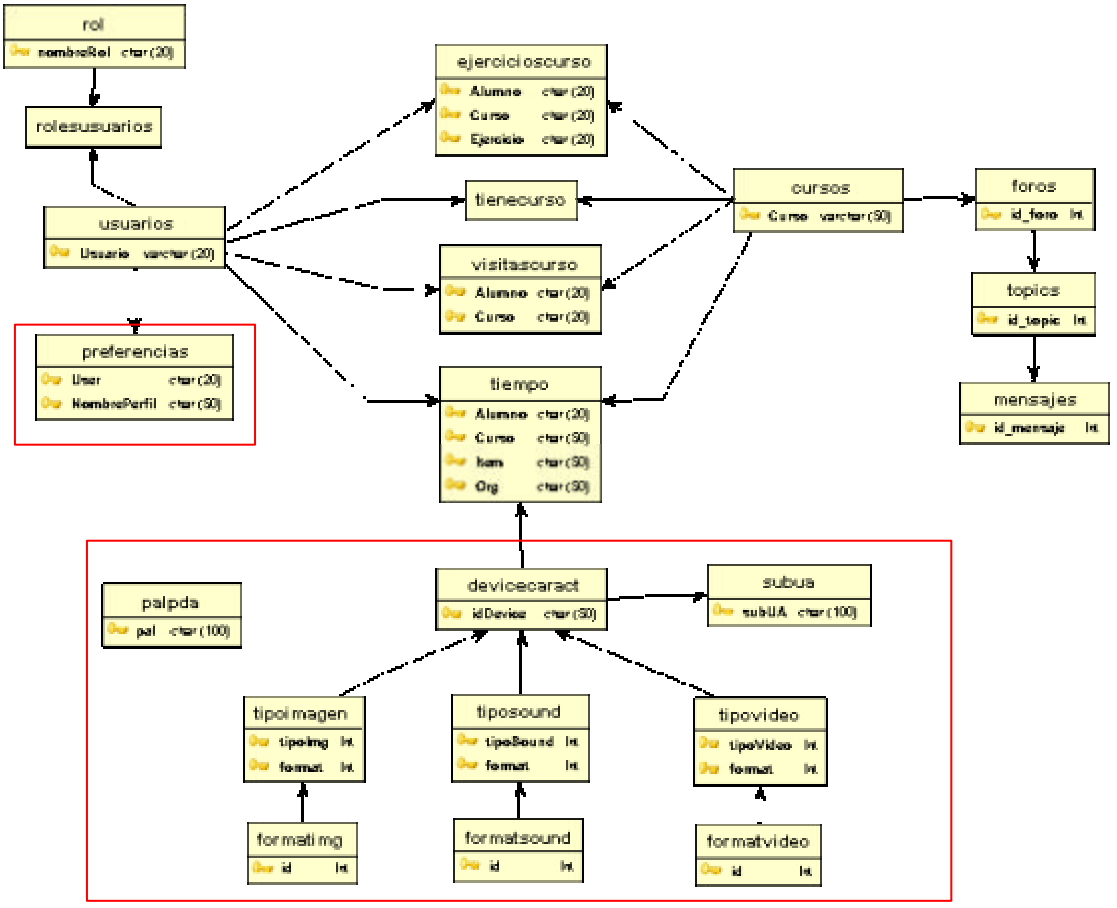
Para ello el sistema e-Aula será envuelto por un wrapper el cual, si se tuviera que modificar alguna de las tablas de la base de datos, lo haga a través de la interfaz.




El interfaz de eventos se ha diseñado mediante una factoría abstracta para no limitar la funcionalidad del mismo

## 9.- Base de datos del sistema. Modificaciones.

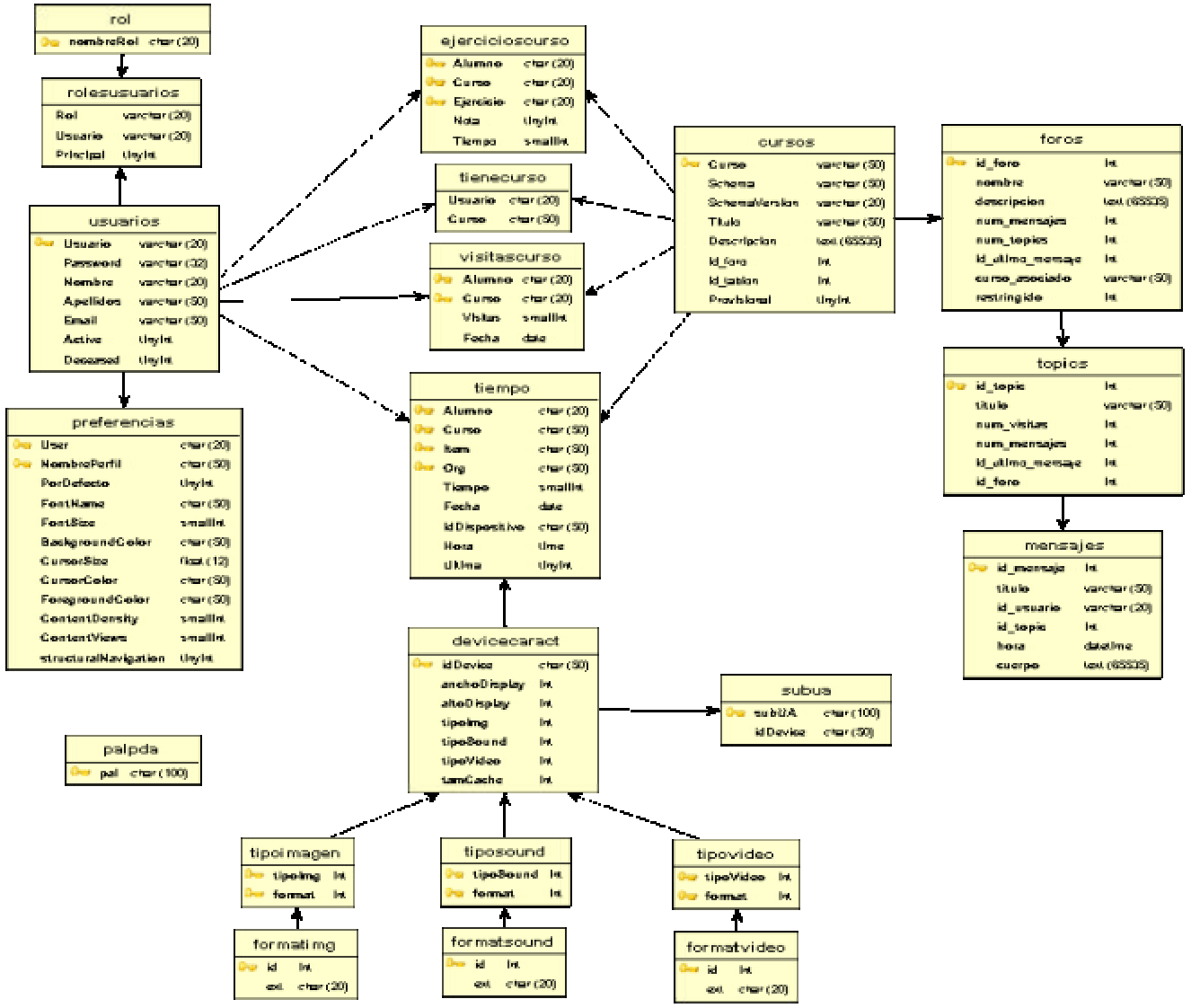
Se han incorporado nuevas tablas al sistema, además de modificadas algunas de ellas. A continuación se muestra un esquema de la base de datos con las tablas involucradas en el módulo implementado en el presente proyecto.




En rojo se encuadran las tablas que se han añadido nuevas en el sistema. Además se ha modificado la tabla tiempo, añadiendo el identificador del tipo de dispositivo que se utilizó en dicha conexión.

En el apéndice b del presente documento se muestran los scripts sql de creación de las tablas nuevas incorporadas al sistema, así como de la tabla tiempos, con la modificación añadida.

A continuación se muestra el esquema completo de la base de datos utilizada en el proyecto.


rol
nombreRol char(20)
rolesusuarios
Rol varchar(20)
Usuario varchar(20)
Principal tinyint
usuarios
Usuario varchar(20)
Password varchar(32)
Nombre varchar(20)
Apellidos varchar(30)
Email varchar(30)
Active tinyint
Deceased tinyint
preferencias
User char(20)
NombrePerfil char(30)
PorDefecto tinyint
FontName char(30)
FontSize smallint
BackgroundColor char(30)
CursorSize float(12)
CursorColor char(30)
ForegroundColor char(30)
ContentDensity smallint
ContentViews smallint
structuralNavigation tinyint
palpda
pal char(100)
ejerciciosourso
Alumno char(20)
Curso char(20)
Ejercicio char(20)
Nota tinyint
Tiempo smallint
tienecurso
Usuario char(20)
Curso char(30)
visitascurso
Alumno char(20)
Curso char(20)
Visitas smallint
Fecha date
tiempo
Alumno char(20)
Curso char(30)
Item char(30)
Org char(30)
Tiempo smallint
Fecha date
IdDispositivo char(30)
Hora time
Ultima tinyint
devicecaract
idDevice char(30)
anchoDisplay int
altoDisplay int
tipoImg int
tipoSound int
tipoVideo int
tamCache int
cursos
Curso varchar(30)
Schema varchar(30)
SchemaVersion varchar(20)
Titulo varchar(30)
Descripcion text(65535)
Id_foro int
Id_tablon int
Provisional tinyint
subua
subUA char(100)
idDevice char(30)
tipoimagen
tipoImg int
format int
formatimg
id int
ext char(20)
tiposound
tipoSound int
format int
formatsound
id int
ext char(20)
tipovideo
tipoVideo int
format int
formatvideo
id int
ext char(20)
foros
id_foro int
nombre varchar(30)
descripcion text(65535)
num_mensajes int
num_topics int
id_ultimo_mensaje int
curso_asociado varchar(30)
restringido int
topics
id_topic int
titulo varchar(30)
num_visitas int
num_mensajes int
id_ultimo_mensaje int
id_foro int
mensajes
id_mensaje int
titulo varchar(30)
id_usuario varchar(20)
id_topic int
hora datetime
cuerpo text(65535)

# Sección VI. Resultados y conclusiones

# 1.- Resultado final del proyecto.

Como resultado del proyecto hemos logrado implementar un módulo que añade distintas funcionalidades al sistema <e-Aula> (funcionalidades descritas a lo largo del presente documento).

El objetivo del proyecto consistía en proponer un ejemplo de uso del estándar de IMS Learner Information Package, añadir distintas funcionalidades al sistema y abordar el problema de la accesibilidad, tanto en términos de preferencias personales como de diferencias entre posibles dispositivos.

Bajo nuestro punto de vista se ha cumplido con la implementación de las nuevas funcionalidades y se ha iniciado un nuevo camino en la adaptación de los contenidos educativos teniendo en cuenta las distintas preferencias de cada persona y las características del dispositivo con el cual acceden al sistema.

Podemos enumerar los siguientes objetivos como cumplidos:

- Queda realizado un exhaustivo estudio del estándar de IMS-LIP y se ofrece una posibilidad de implementación a la hora de importar y exportar la información de los alumnos. Añadida la extensión ACCLIP, se consigue incorporar las características relativas a las preferencias de accesibilidad del alumno.

- Se ha añadido la idea de “perfil”, término que engloba las preferencias de los alumnos a la hora de acceder al sistema, pudiendo dichos alumnos editar estos perfiles.

- Quedan añadidas las nuevas funcionalidades relativas a la sección de estadísticas del sistema.

- Se ha iniciado el camino hacia la detección del dispositivo de conexión. Hemos cimentado el comienzo de un módulo (totalmente ampliable) cuya finalidad consiste en poder detectar

con qué tipo de dispositivo está accediendo el alumno al sistema.

- En base a las preferencias de los alumnos y a la detección del dispositivo, se ha tratado de incorporar al sistema la adaptación de los contenidos educativos a cada uno de los usuarios concretos en cada momento y en cada situación.

El proyecto ha sido abordado teniendo en cuenta siempre que existe un sistema principal que en un futuro deberá incorporar las funcionalidades que hemos estudiado y desarrollado. Por ello en nuestro empeño ha primado la modularidad y, sobre todo, la flexibilidad del diseño (iniciar un camino que deberá ser retomado en un futuro para la total o parcial incorporación de las ideas aportadas en el sistema original).

## 2.- Habilidades adquiridas.

El afrontar un proyecto de estas características te obliga a profundizar en distintos ámbitos de tecnologías actualmente en auge. Es una de la principales ventajas de desarrollar un proyecto que pretende utilizar las últimas tecnologías existentes. Es por ello que gran parte del proyecto haya sido dedicado al estudio de diferentes estándares e-learning y tecnologías tales como Struts o XML.

Se ha adquirido una útil experiencia en cuanto a la forma de enfrentarse a un estándar. No resulta fácil acometer este tipo de tareas. Un estándar suele ser escrito pensando en que el lector tendrá un amplio dominio del entorno, y suelen ser largos, pesados y, normalmente, redactados en un inglés muy técnico.

Sin duda la experiencia de este proyecto nos ayudará a afrontar este tipo de tareas (muy común en la profesión) estando más cualificados para su comprensión.

Hemos avanzado conocimientos sobre la plataforma J2EE, iniciándonos en Jakarta Struts, una tecnología no demasiado extendida actualmente.

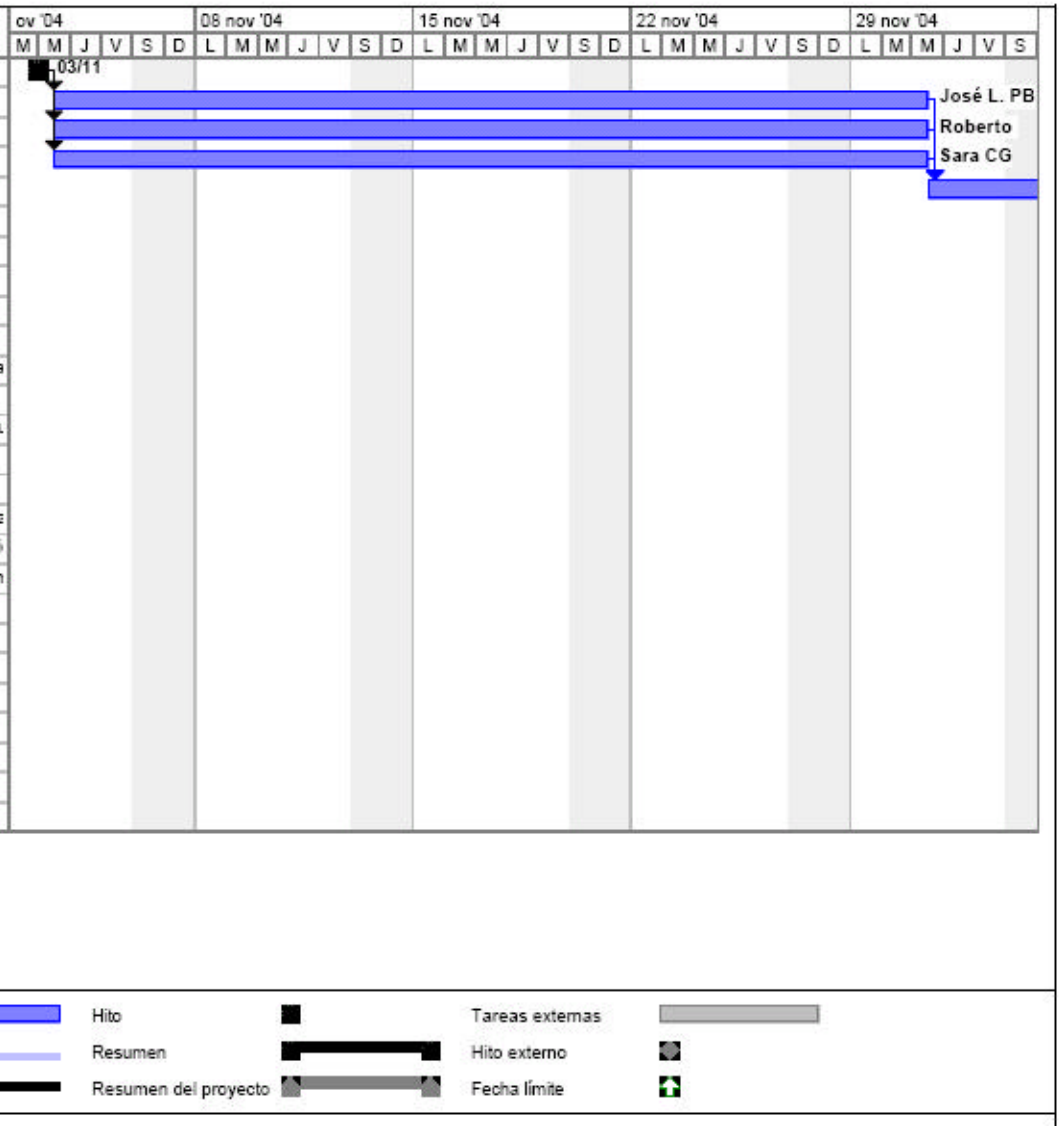
Sin duda hemos desarrollado capacidades en la programación con XML, comprendiendo las posibilidades que nos ofrece el leguaje de marcado. El uso conjunto de Java y XML, las posibilidades que nos ofrece XSL, etc.

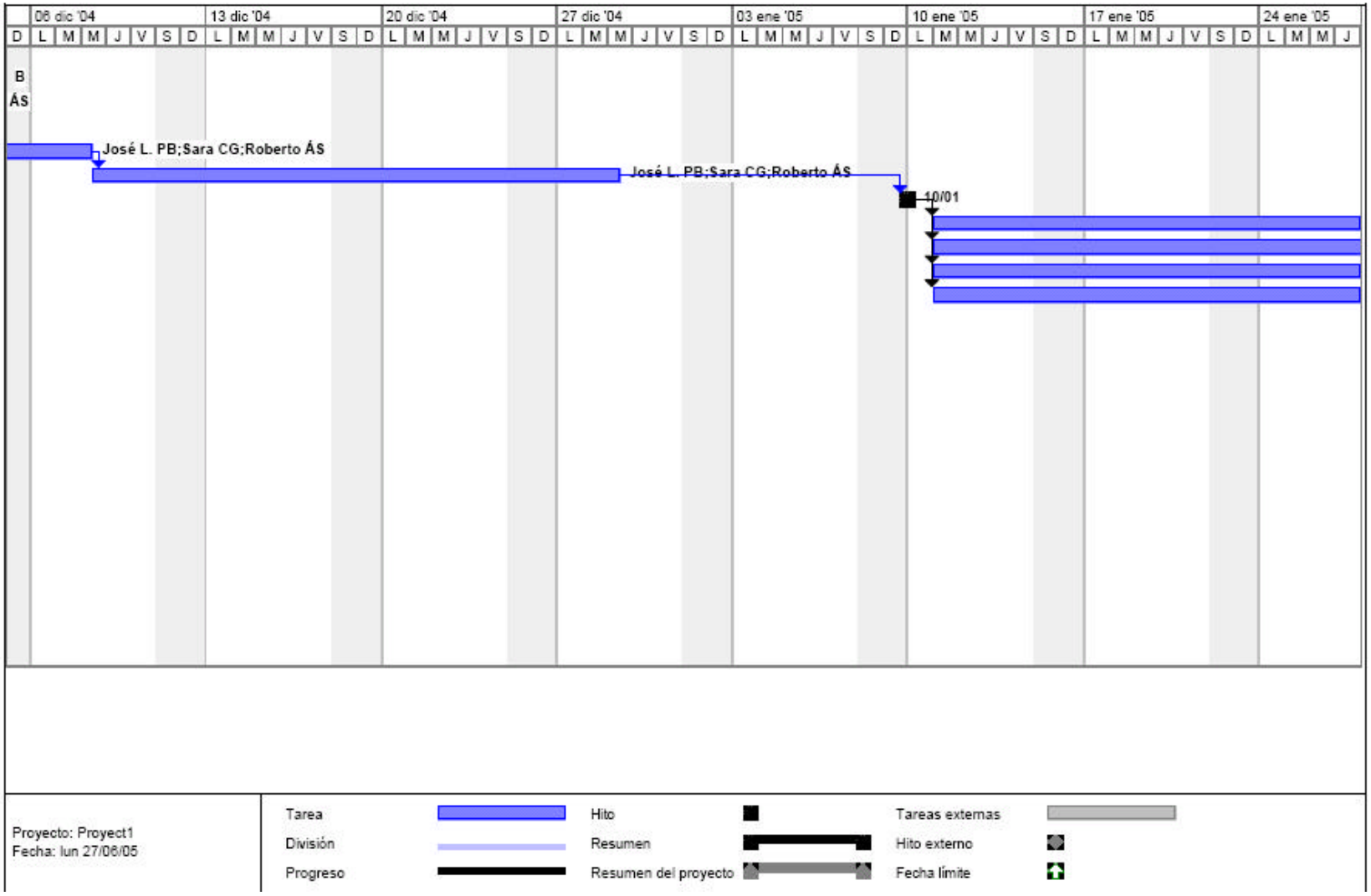
Actualmente el campo de la accesibilidad se está imponiendo en los diseños. Nosotros en este proyecto hemos abordado el tema y hemos comprendido los motivos que invitan a la utilización de tecnologías que ofrecen posibilidades de accesibilidad.

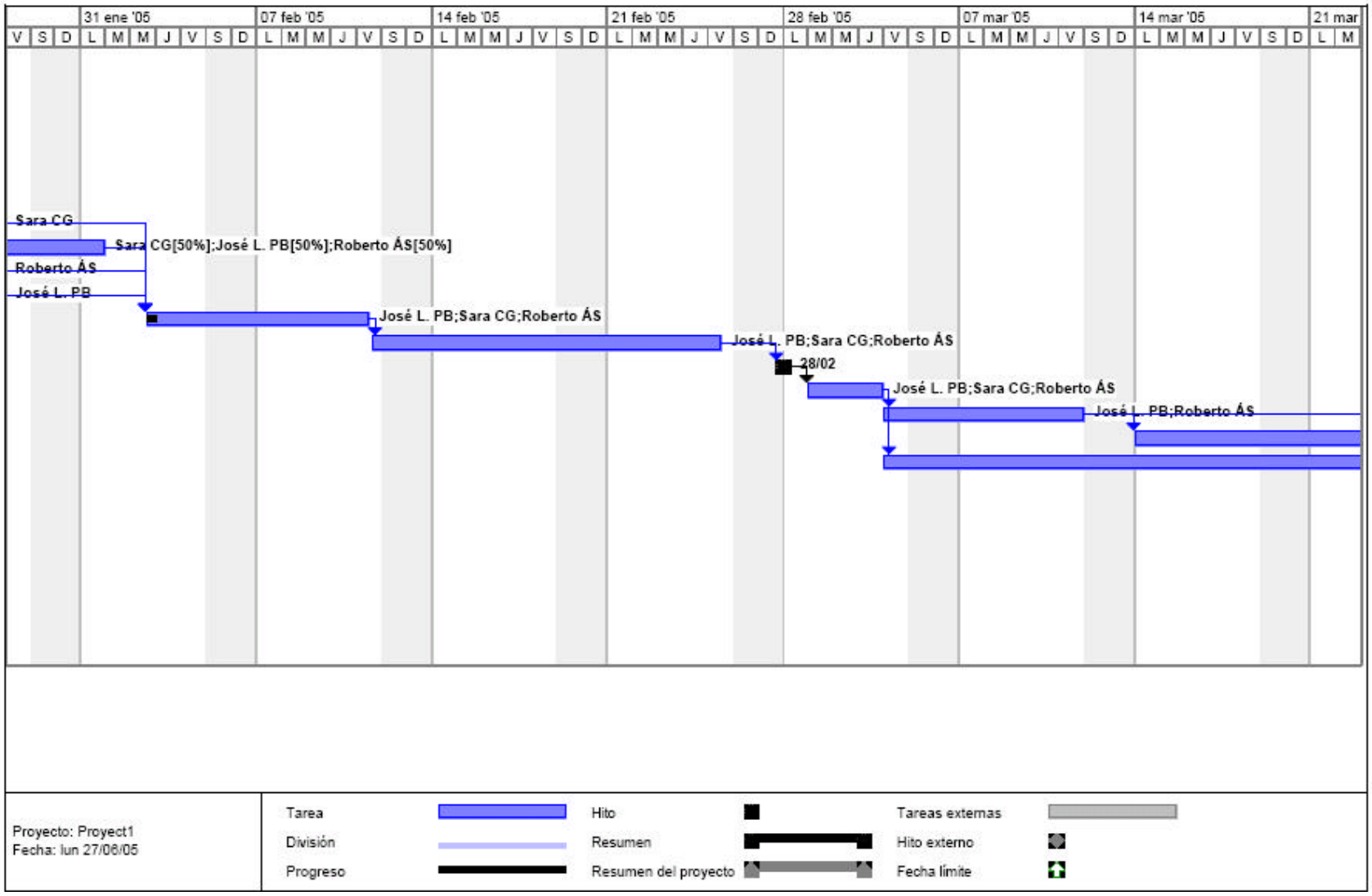
Aunque se aparte del tema de las tecnologías, también cabe destacar la experiencia adquirida en el trabajo en equipo. Hemos conseguido una experiencia que pensamos nos beneficiará en un futuro a la hora de afrontar trabajos en los que debe verse implicado un grupo de personas, no personas concretas. Este ha sido probablemente el trabajo más “realista” que hemos realizado los miembros de este grupo hasta la fecha, afrontando los problemas que surgen en el día a día en las empresas de desarrollo de soluciones informáticas.

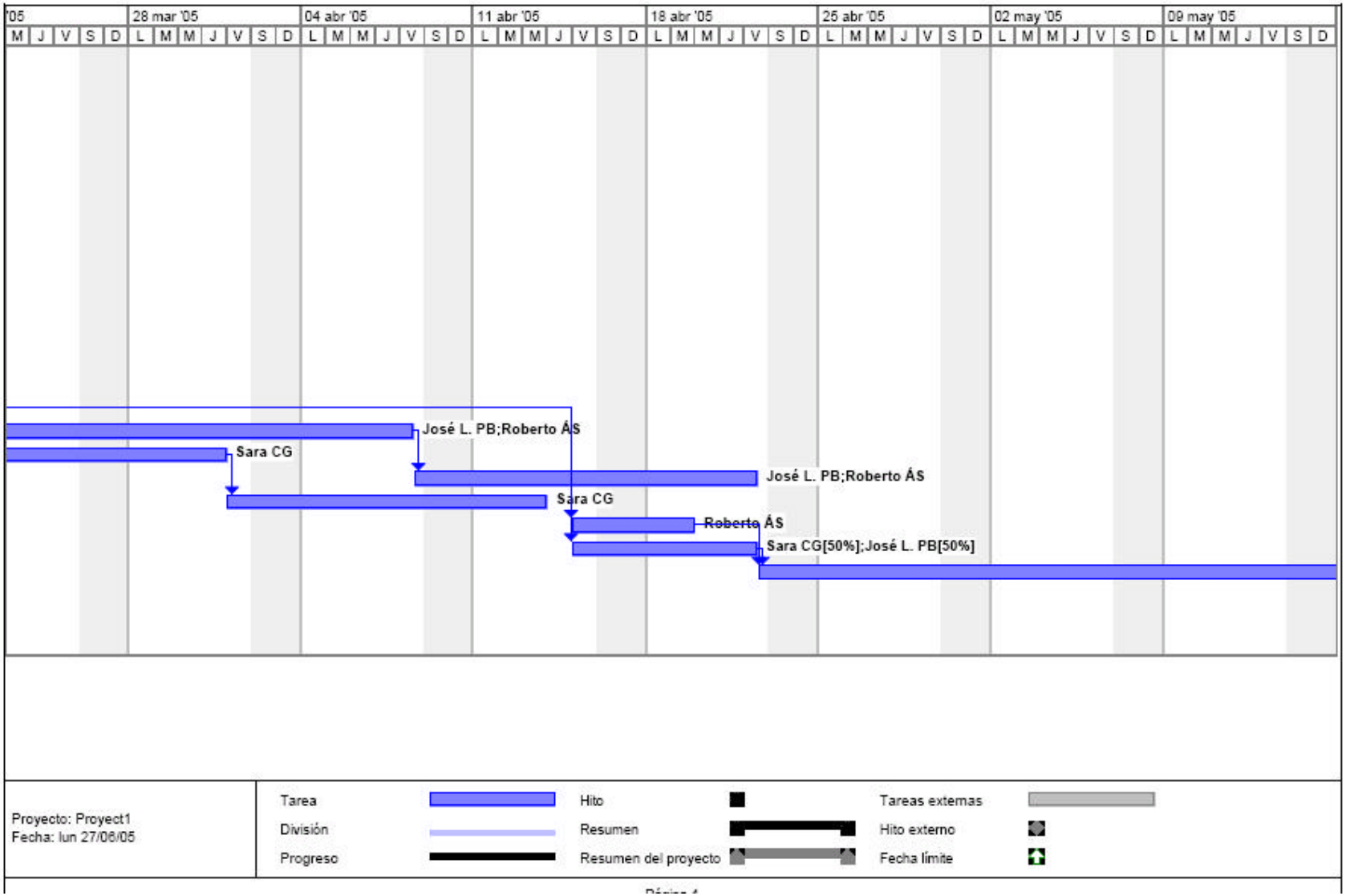
# Sección VII. Apéndice a

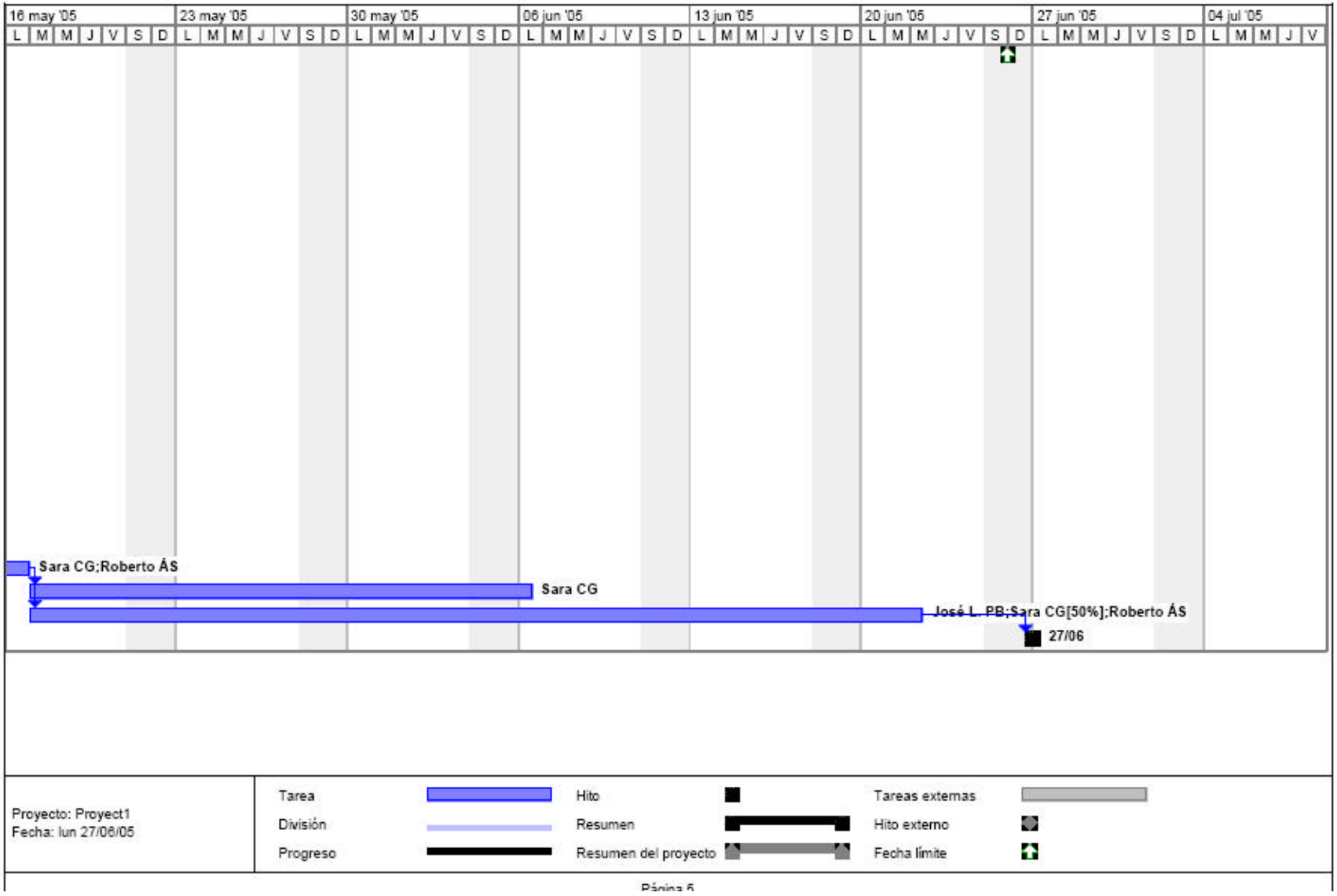
| Id | Nombre de tarea |
|---|---|
| 1 | Reunión con el cliente 1 |
| 2 | Estudio del estandar ADL-ACORM 2004 |
| 3 | Estudio del estándar LIP de IMS |
| 4 | Estudio de la tecnología Jakarta Struts |
| 5 | Revisión del trabajo desarrollado |
| 6 | Redacción de un documento descriptivo de los estudios |
| 7 | Reunión con el cliente 2 |
| 8 | Implementación de un ejemplo del modelo Struts |
| 9 | Estudio del estándar ACCLIP de IMS |
| 10 | Revisión y corrección de puntos concretos del documento |
| 11 | Implementación de un ejemplo de modelo SCORM basado en la |
| 12 | Revisión y redacción final del documento |
| 13 | Implicación de todos los miembros en todos los puntos del estu |
| 14 | Reunión con el cliente 3 |
| 15 | Estudio de los casos de uso del sistema |
| 16 | Estudio de los campos LIP-ACCLIP a utilizar. Diseño de la tabla |
| 17 | Análisis e implementación del módulo de importación exportaci |
| 18 | Análisis e implementación del módulo de estadísticas del sisten |
| 19 | Pruebas Import/Export |
| 20 | Pruebas Estadística |
| 21 | Diseño de tablas para almacenar información de dispositivos |
| 22 | Módulo detector de dispositivo |
| 23 | Transformación de la presentación del contenido |
| 24 | Pruebas de la tranformación de la presentación |
| 25 | Redacción del documento final descriptivo del proyecto |
| 26 | Entrega final |




06 dic '04
13 dic '04
20 dic '04
27 dic '04
03 ene '05
10 ene '05
17 ene '05
24 ene '05
D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J
B
ÁS
José L. PB;Sara CG;Roberto ÁS
José L. PB;Sara CG;Roberto ÁS
10/01
Proyecto: Proyect1
Fecha: lun 27/06/05
Tarea
División
Progreso
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Fecha límite

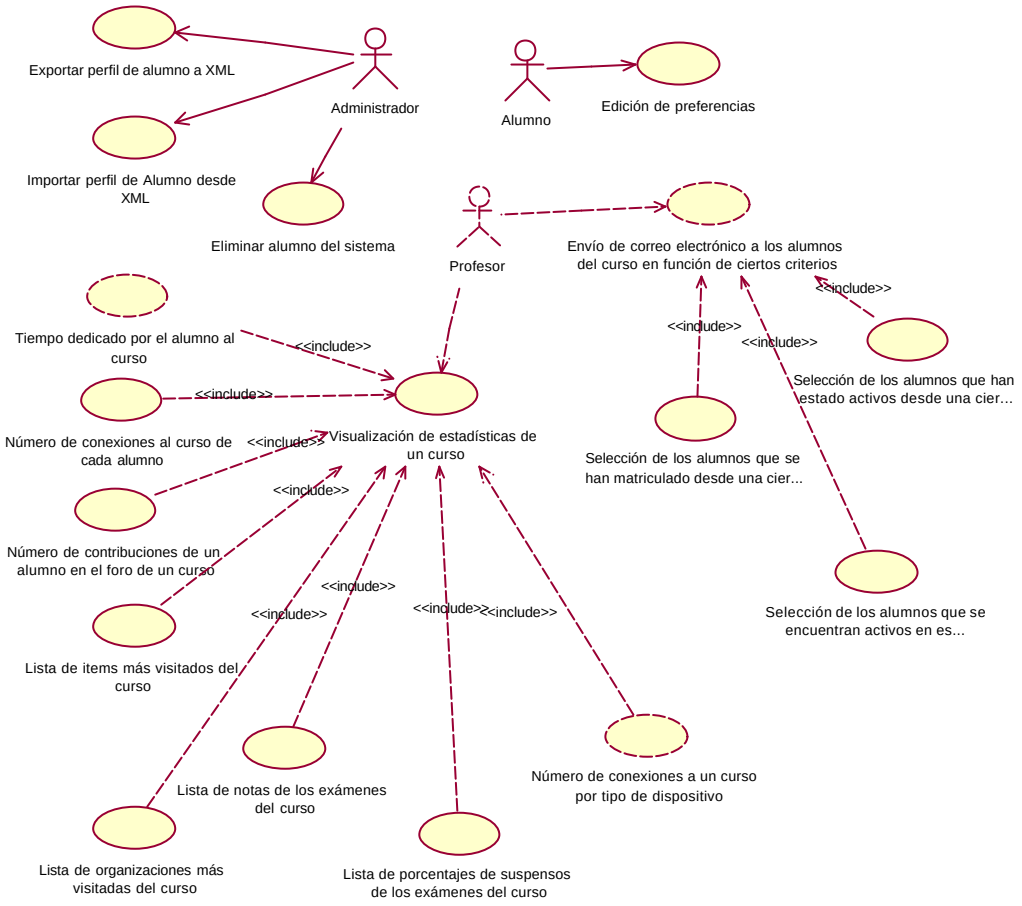

31 ene '05
07 feb '05
14 feb '05
21 feb '05
28 feb '05
07 mar '05
14 mar '05
21 mar
Sara CG
Sara CG[50%];José L. PB[50%];Roberto ÁS[50%]
Roberto ÁS
José L. PB
José L. PB;Sara CG;Roberto ÁS
José L. PB;Sara CG;Roberto ÁS
28/02
José L. PB;Sara CG;Roberto ÁS
José L. PB;Roberto ÁS
Proyecto: Proyect1
Fecha: lun 27/06/05
Tarea
División
Progreso
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Fecha límite


'05
28 mar '05
04 abr '05
11 abr '05
18 abr '05
25 abr '05
02 may '05
09 may '05
M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D
José L. PB;Roberto ÁS
Sara CG
José L. PB;Roberto ÁS
Sara CG
Roberto ÁS
Sara CG[50%];José L. PB[50%]
Proyecto: Proyect1
Fecha: lun 27/06/05
Tarea
División
Progreso
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Fecha límite


16 may '05
23 may '05
30 may '05
06 jun '05
13 jun '05
20 jun '05
27 jun '05
04 jul '05
L M M J V S D
Sara CG;Roberto ÁS
Sara CG
José L. PB;Sara CG[50%];Roberto ÁS
27/06
Proyecto: Proyect1
Fecha: lun 27/06/05
Tarea
División
Progreso
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Fecha límite

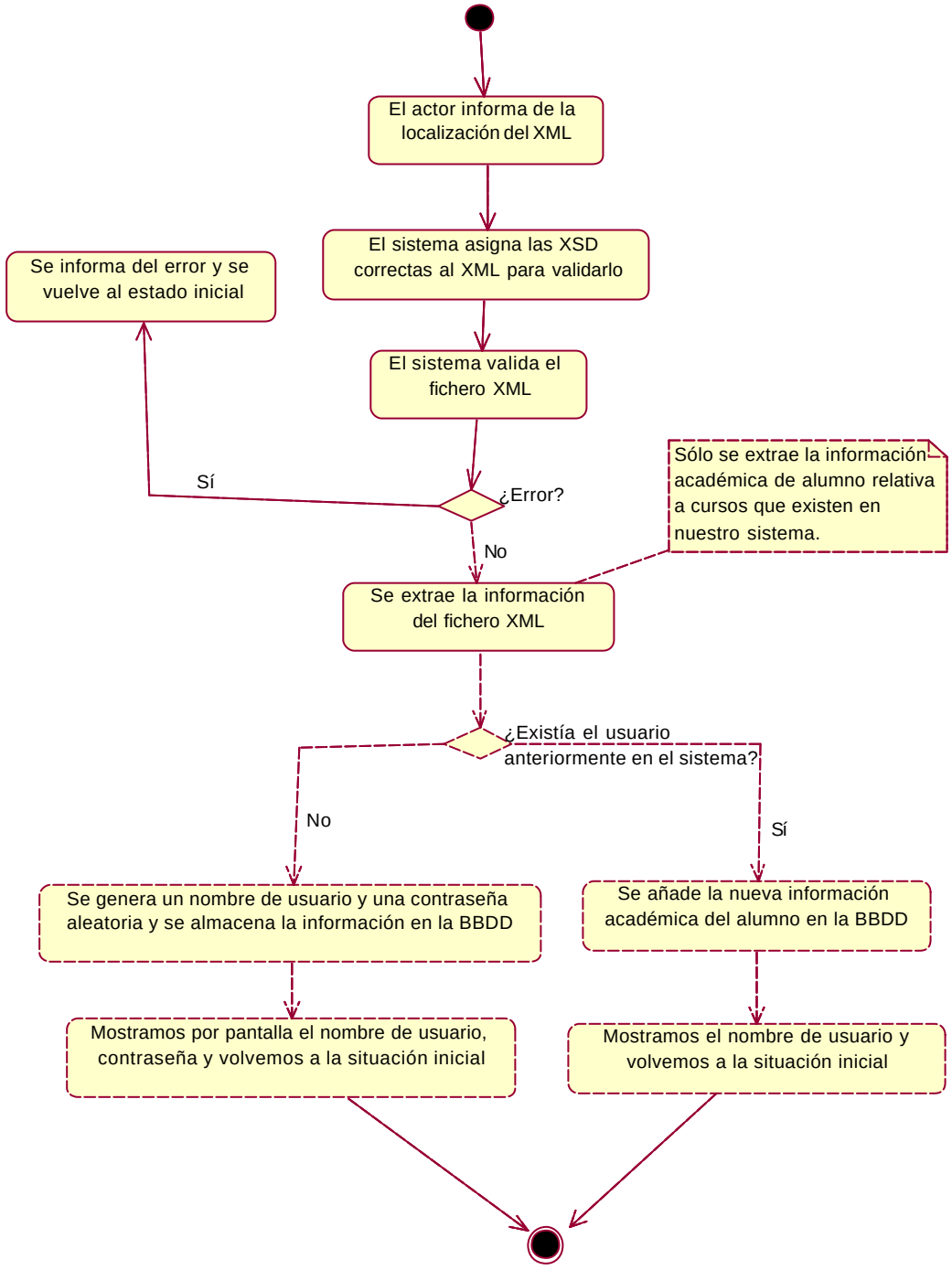
# 1.- Diagramas UML descriptivos del sistema.

## 1.1.- Diagrama de casos de uso

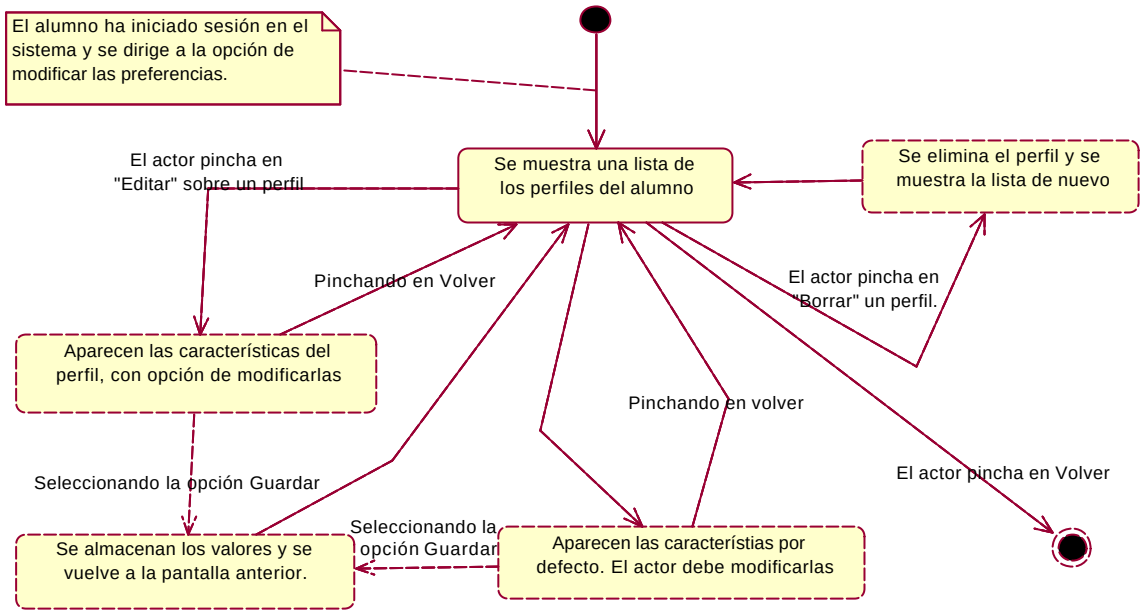


## 1.2.- Diagramas de actividades de los casos de uso.

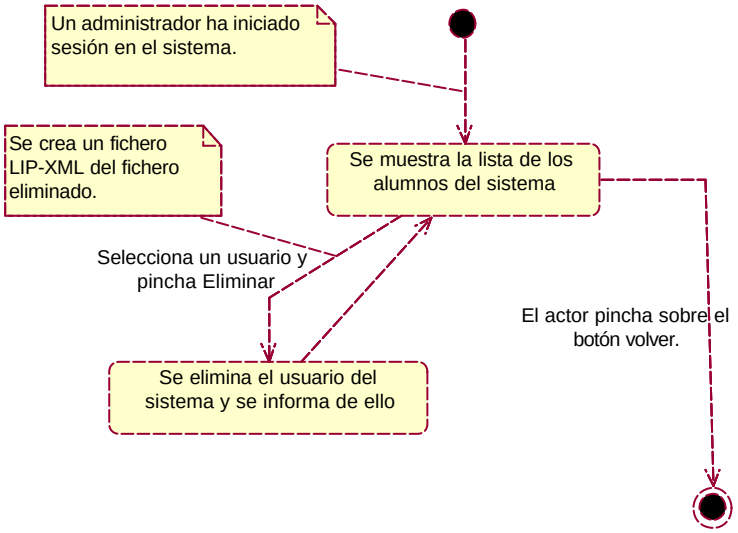
### 1.2.1.- Importación de un fichero XML descriptivo de un alumno.



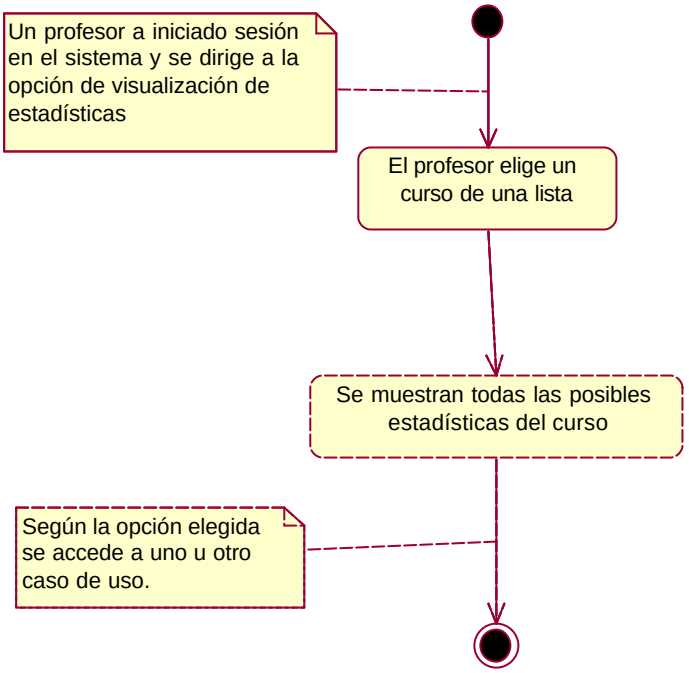
### 1.2.2.- Edición de las preferencias de un Alumno.



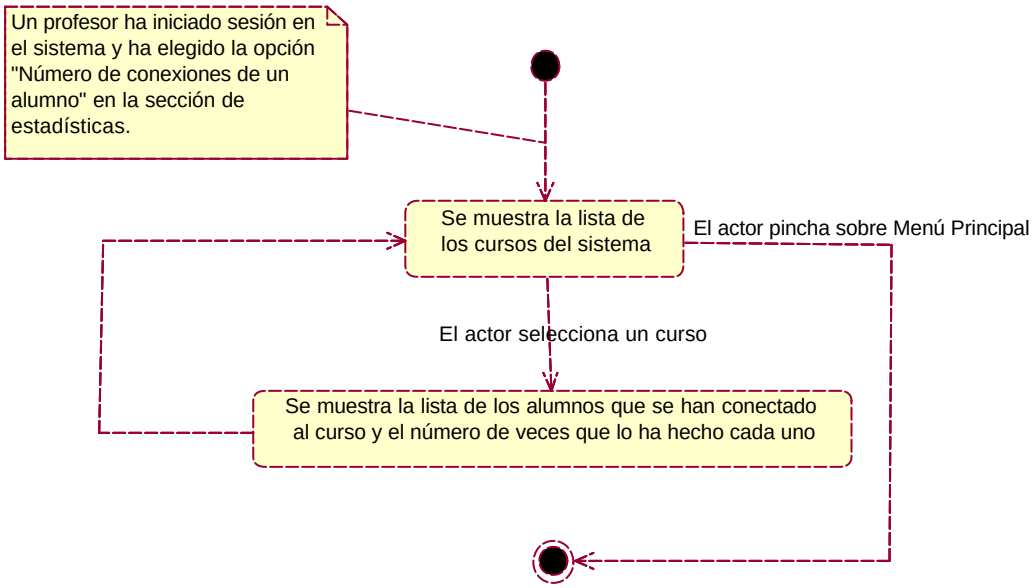

### 1.2.3.- Eliminar un alumno del sistema.



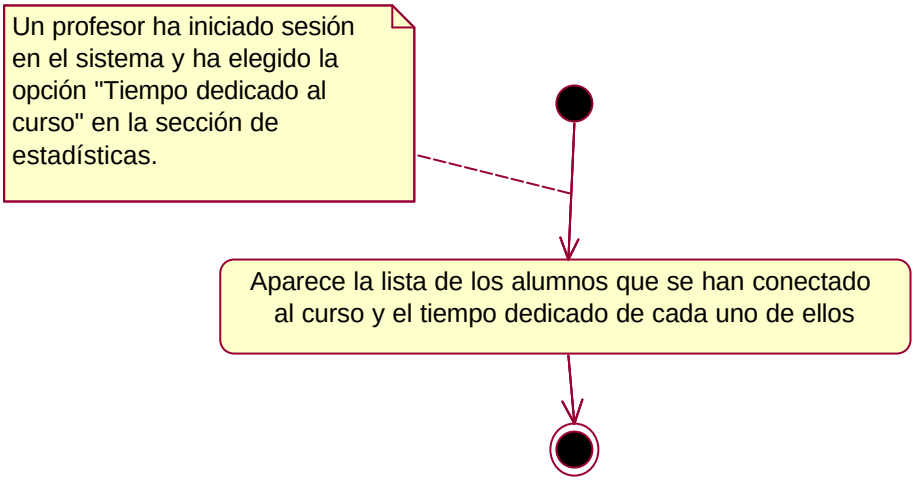
### 1.2.4.- Visualización de estadísticas de un curso.



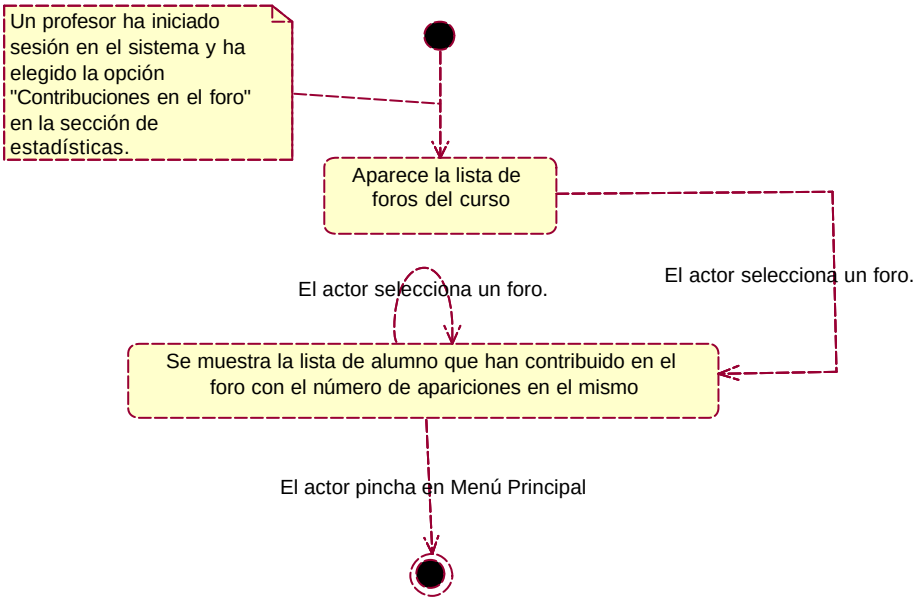

### 1.2.5.- Número de conexiones a un curso.



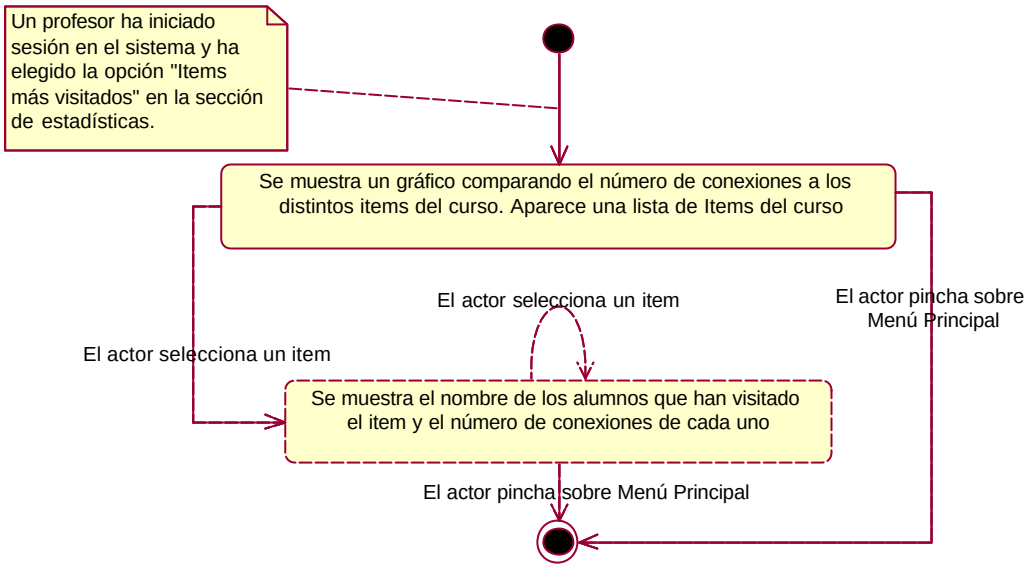
### 1.2.6.- Tiempo dedicado al curso por los alumnos



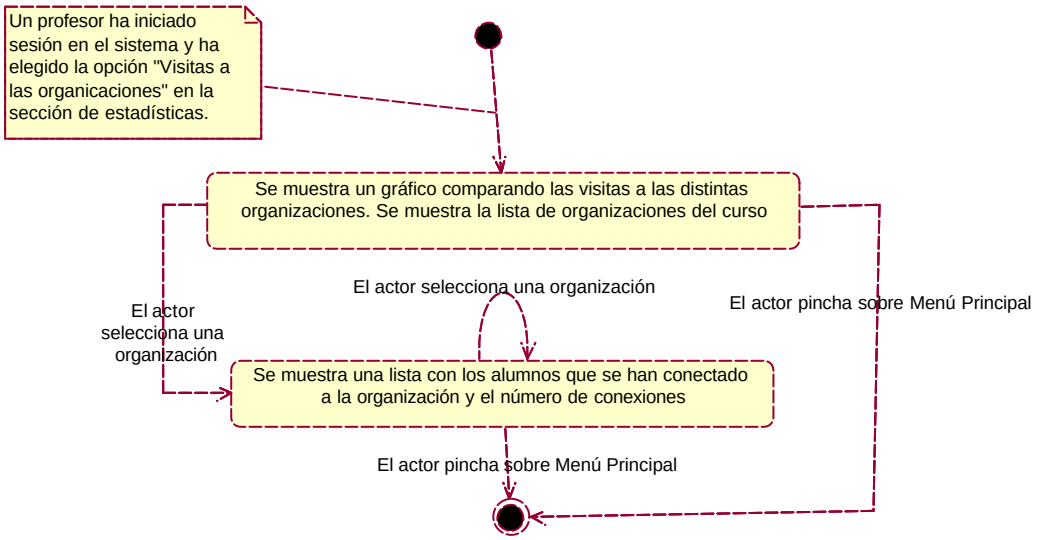

### 1.2.7.- Contribuciones de los alumnos en el foro.



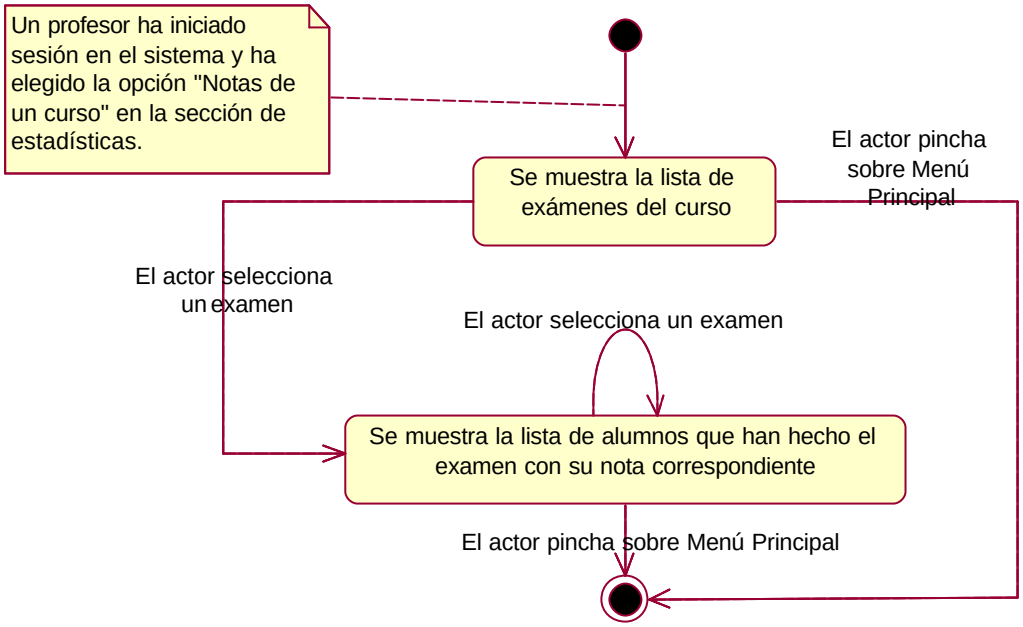
### 1.2.8.- Número de conexiones a los Items del curso.



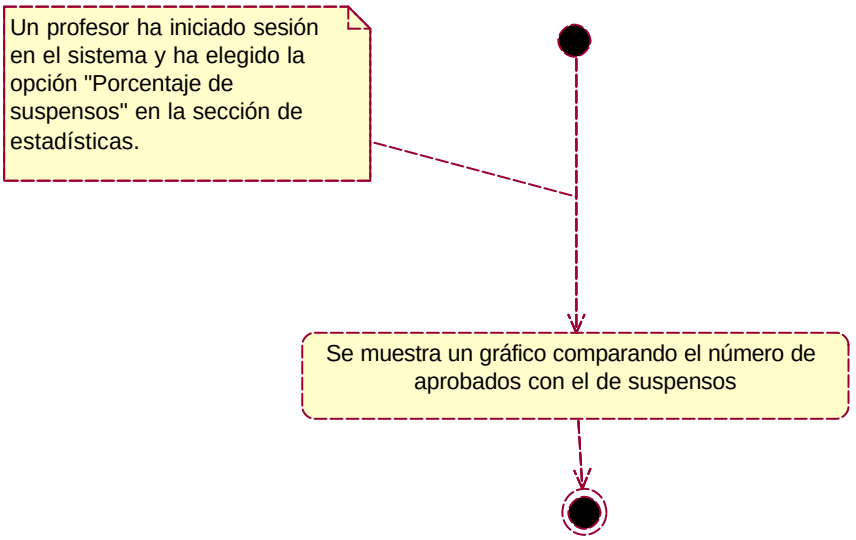

### 1.2.9.- Número de conexiones a las organizaciones de un curso.



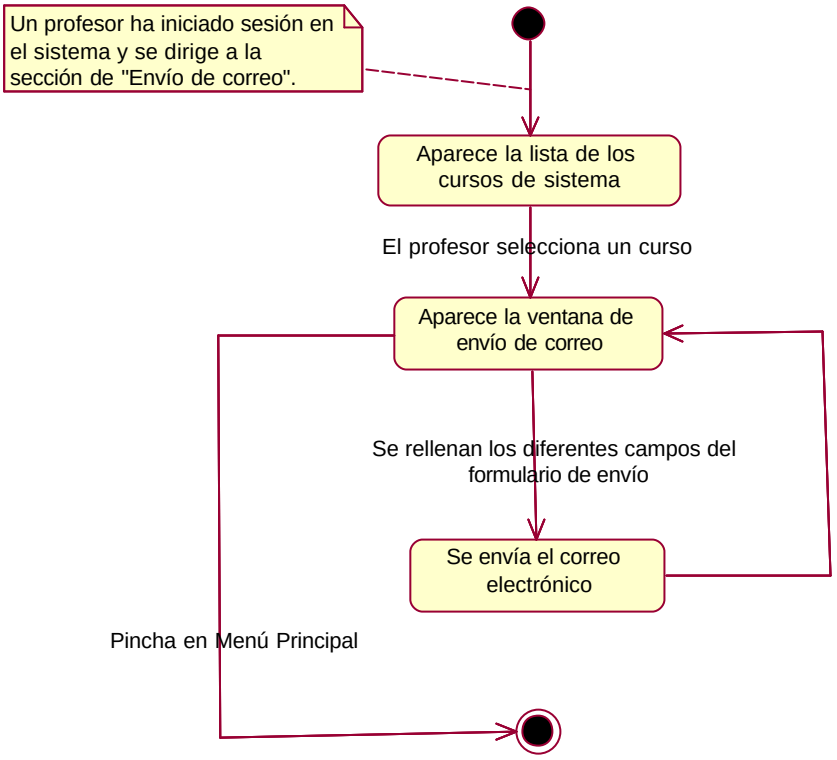
### 1.2.10.- Lista de notas de un curso



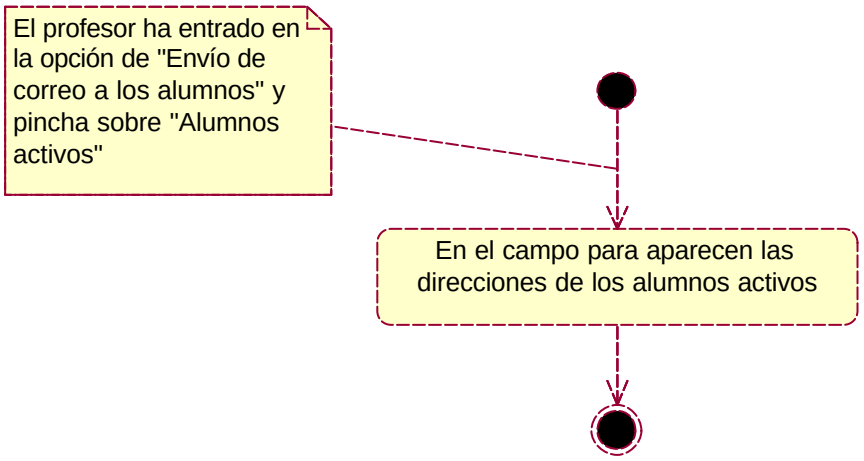

### 1.2.11.- Porcentaje de suspensos de un curso



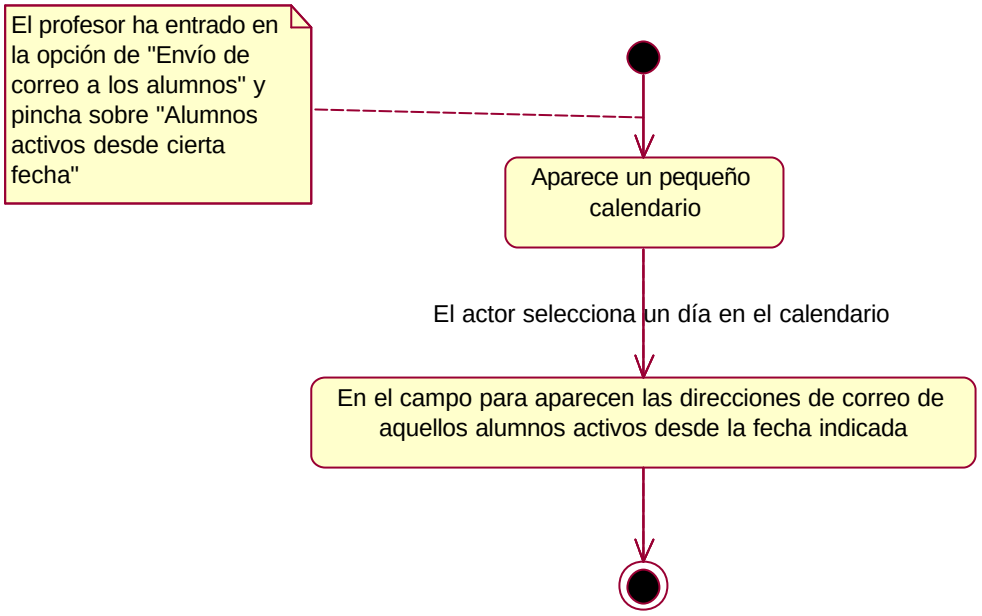
### 1.2.12.- Envío de un correo electrónico a los alumnos de un curso.



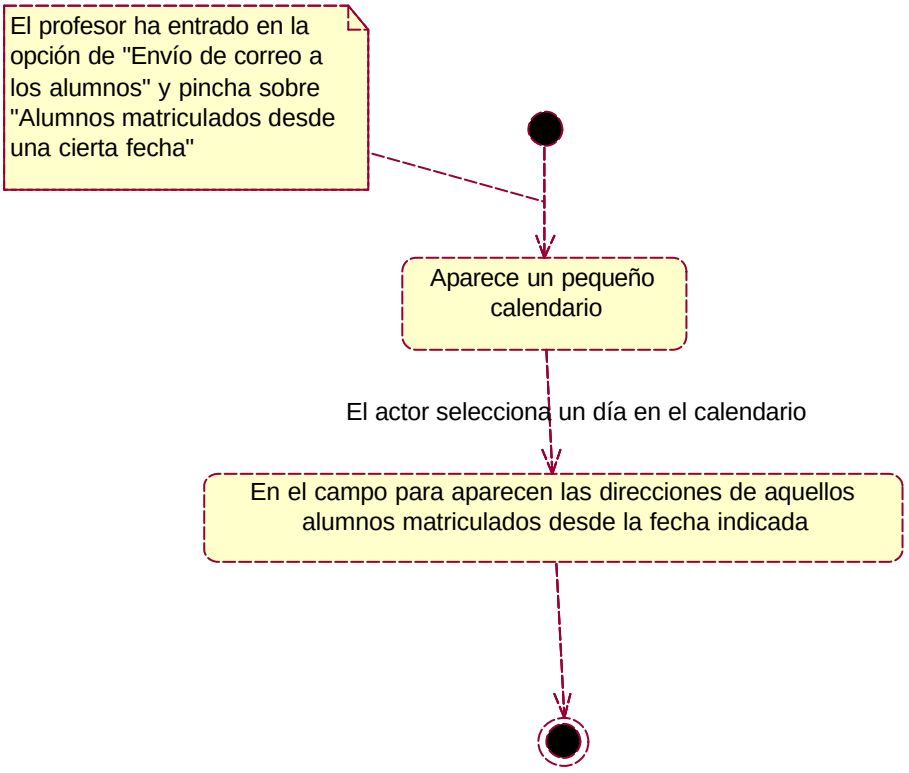

### 1.2.13.- Selección de alumnos activos



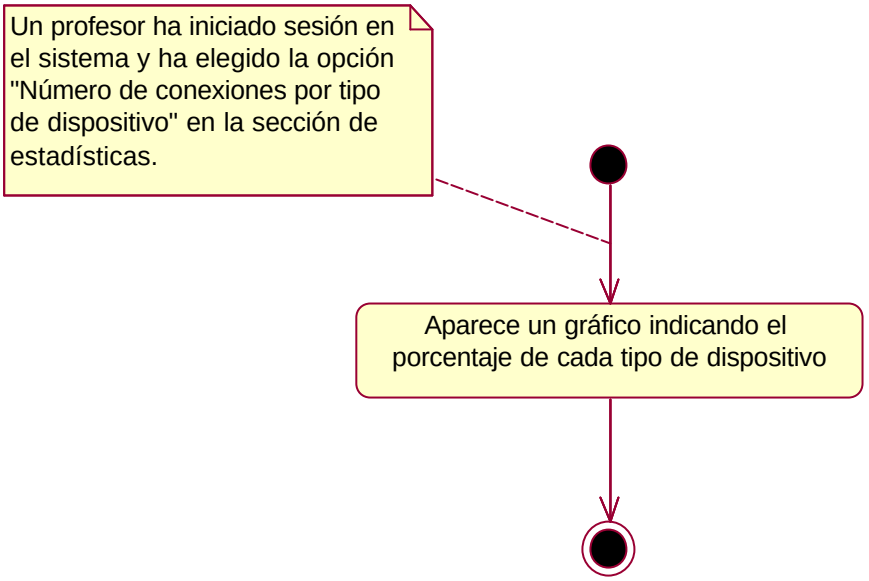
### 1.2.14.- Selección de alumnos desde una cierta fecha.




### 1.2.15.- Selección de alumnos matriculados desde una cierta fecha.



### 1.2.16.- Número de conexiones a un curso por tipo de dispositivo.



# Sección VIII. Apéndice b.

A continuación se detallan algunos ejemplos de uso del programa implementando, mostrando así los resultados de lo anteriormente mostrado.

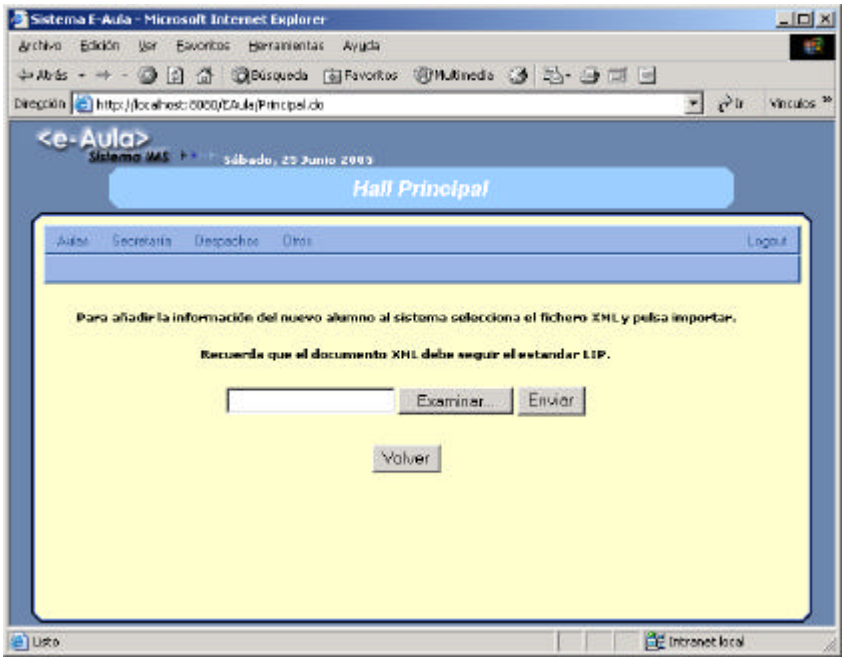
## 1.- Menú Principal.

Al entrar en nuestra aplicación ejemplo aparece un pequeño menú de opciones, desde las cuales se tiene acceso a cada una de las secciones del prototipo.

Desde aquí tenemos acceso a las distintas funcionalidades descritas: Importación, exportación y eliminación de alumnos, estadísticas, envío de correo electrónico y edición de preferencias de un alumno.

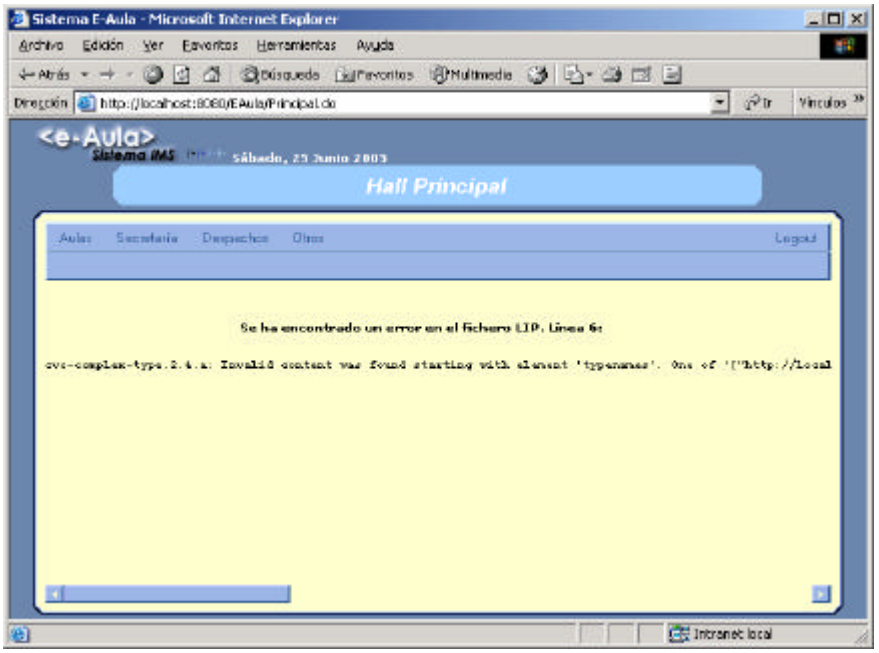
## 2.- Importación/Exportación de XML´s descriptivos de alumnos.

Al acceder a la sección de importación de un alumno se nos ofrece la posibilidad de seleccionar el fichero XML de nuestro equipo.

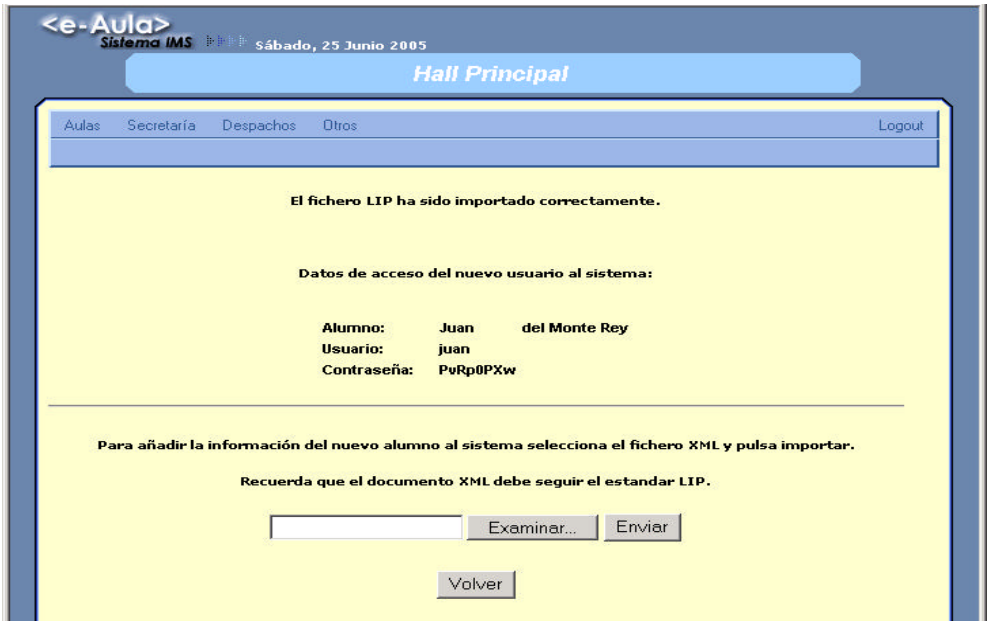



Al pulsar sobre enviar el sistema responde con el resultado de la importación.

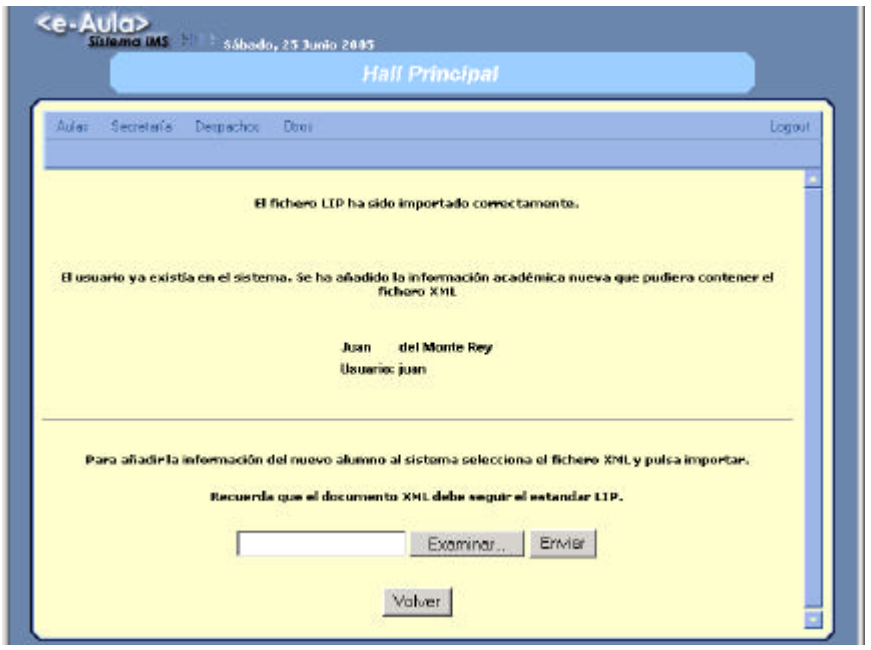
En caso de que el fichero XML no se adapte al estándar LIP, se muestra por pantalla el primero error encontrado en la validación del documento.




Si el formato del documento es correcto pueden ocurrir dos cosas, que el alumno no existiera previamente en el sistema,



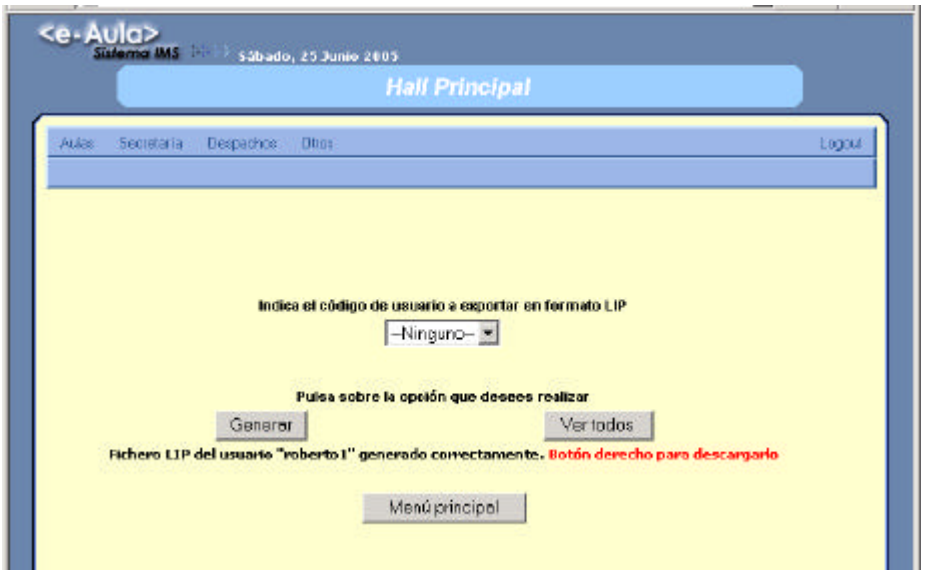
o que ya existiera en el mismo.




En la parte de exportación tenemos una lista con los nombres de usuario de todos los alumnos del curso.

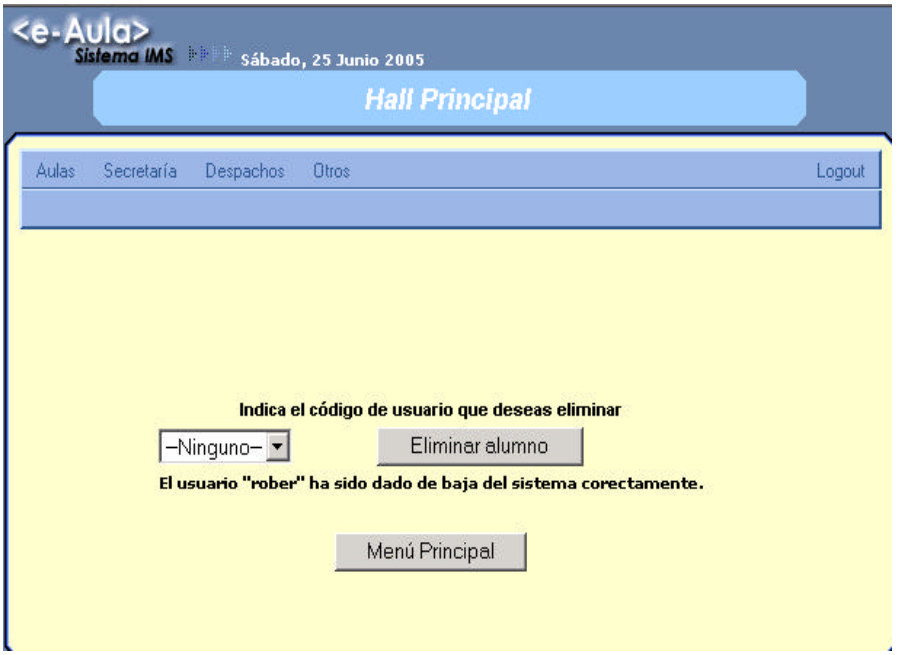
Los ficheros XML generados se almacenan para poder mostrarlos después, por ello tenemos la opción de ver todos los ficheros XML que tenemos almacenados, además de la correspondiente de generar fichero y vuelta al menú.

Cuando generamos el XML se nos muestra un enlace al mismo, desde el cual podemos verlo y/o descargarlo.



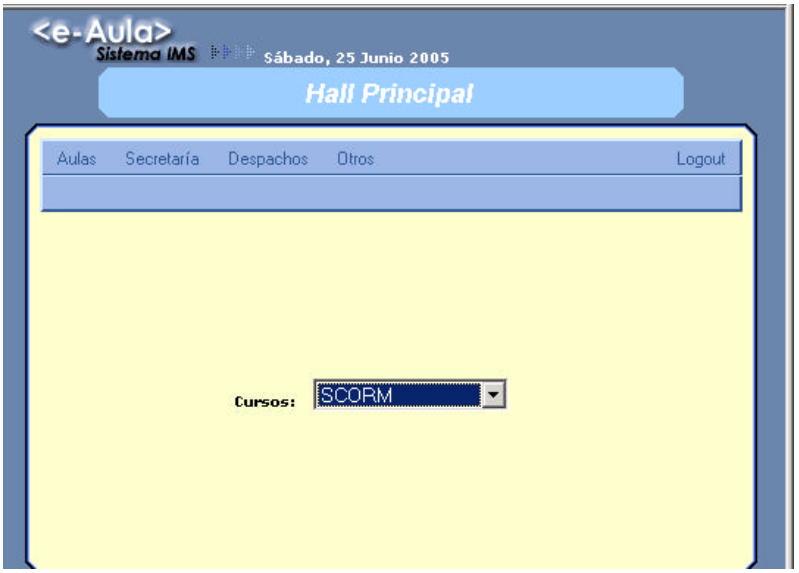

## 3.- Eliminación de un alumno del sistema.

Se nos muestra una lista de los nombres de usuario de los alumnos del sistema y la opción de eliminarlos. Una vez eliminado se informa a través de un mensaje.

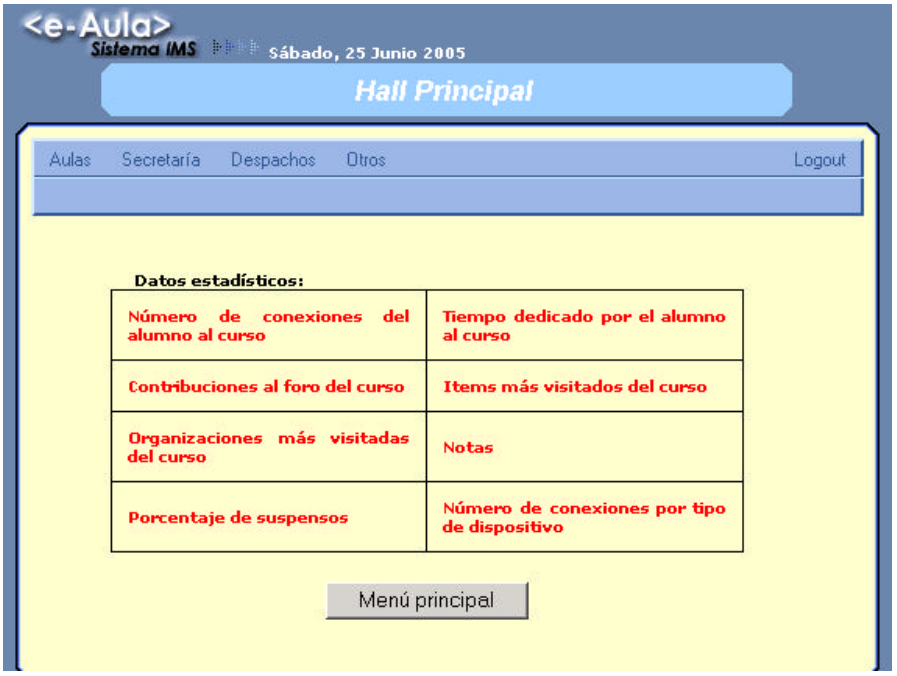


## 4.- Sección de Estadísticas.

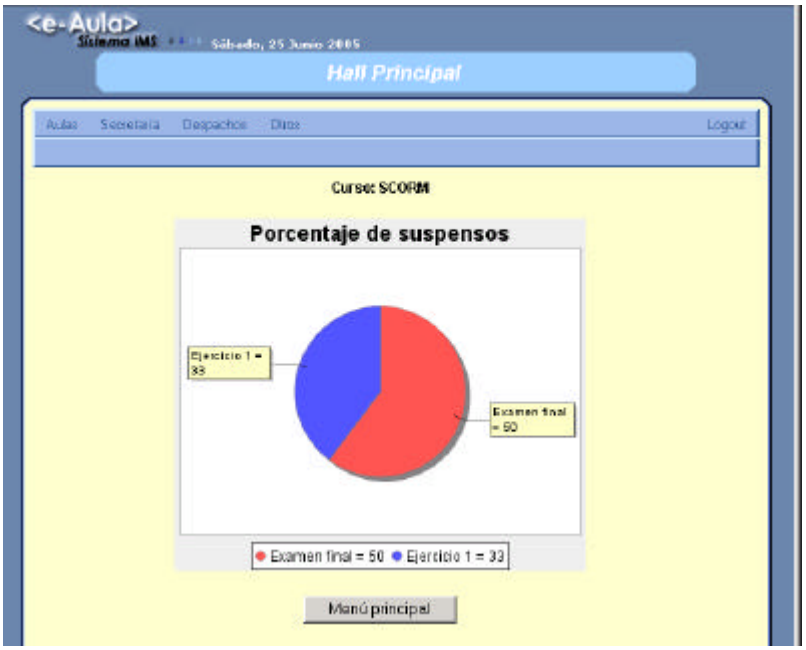
Cuando entramos en la sección de estadísticas lo primero que debemos hacer es seleccionar un curso de la lista de todos los cursos del sistema.




Una vez seleccionado un curso del sistema entramos a una ventana en la que se nos ofrecen todas las posibilidades para mostrar datos estadísticos sobre dicho curso.



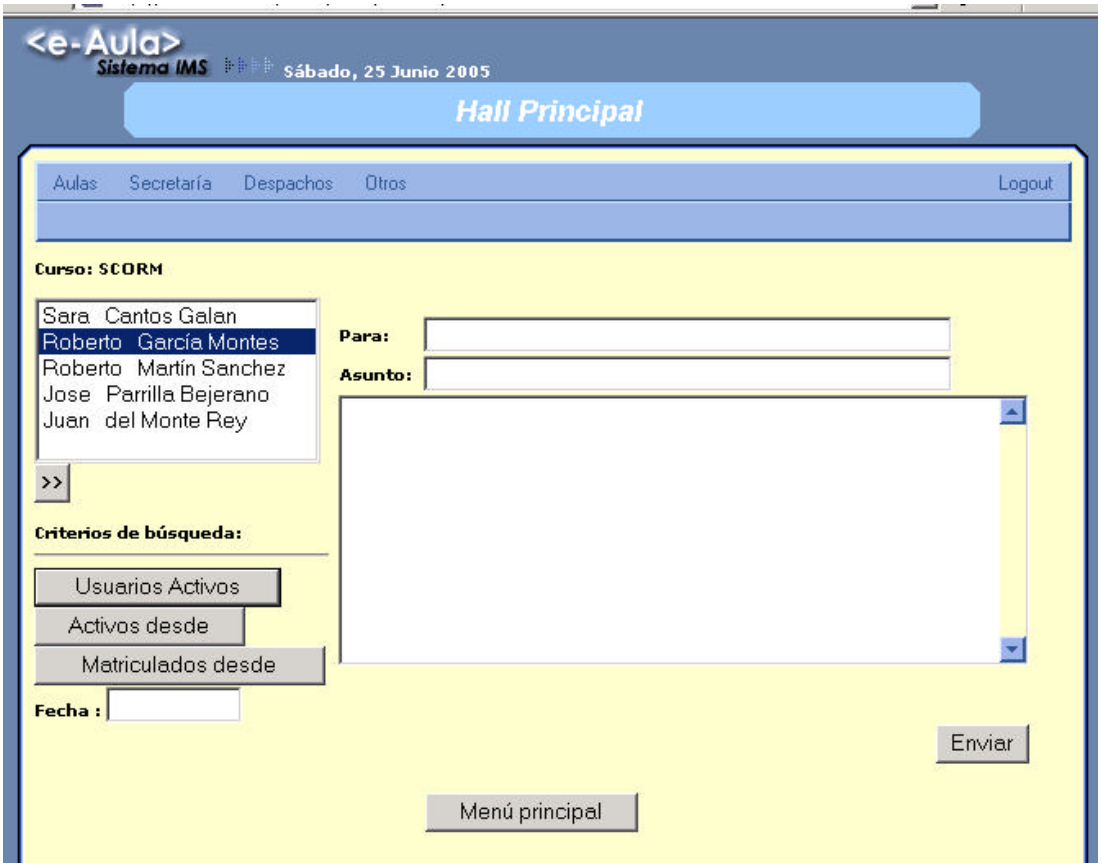
Por ejemplo, seleccionando el curso SCORM, vemos que del total de los suspensos hay un mayor porcentaje del examen final (un 50% frente a un 33% de alumnos que han suspendido el ejercicio 1).




Otro modelo es el que ofrece el apartado "Organizaciones más visitadas". Se nos muestra el gráfico que compara las visitas a las diferentes organizaciones, y además se nos ofrece la posibilidad de seleccionar una organización en concreto para ver el número de conexiones a la misma.

## 5.- Envío de un correo electrónico a los alumnos de un curso.

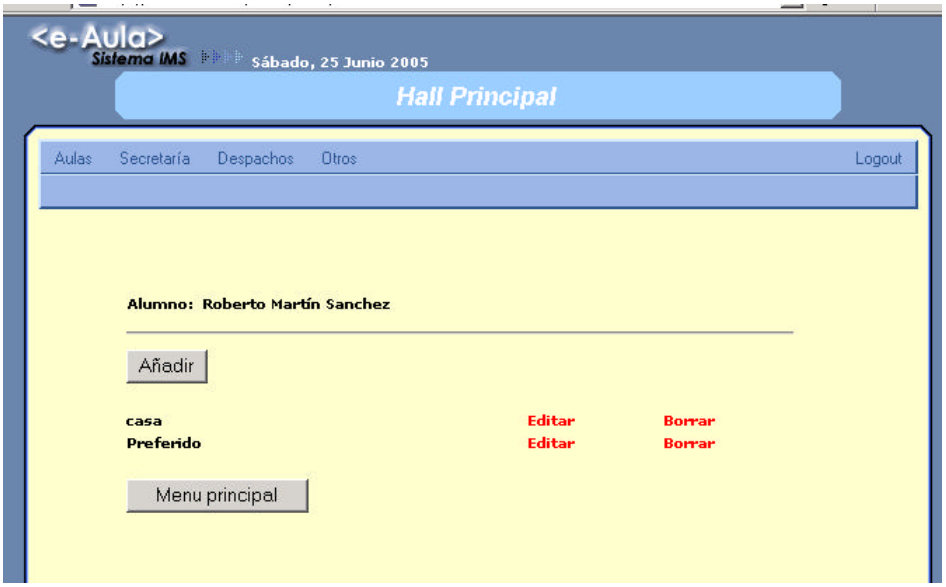
Al entrar en la sección de envío de correo electrónico se nos da a elegir un curso del sistema. Una vez seleccionado aparece la ventana siguiente:



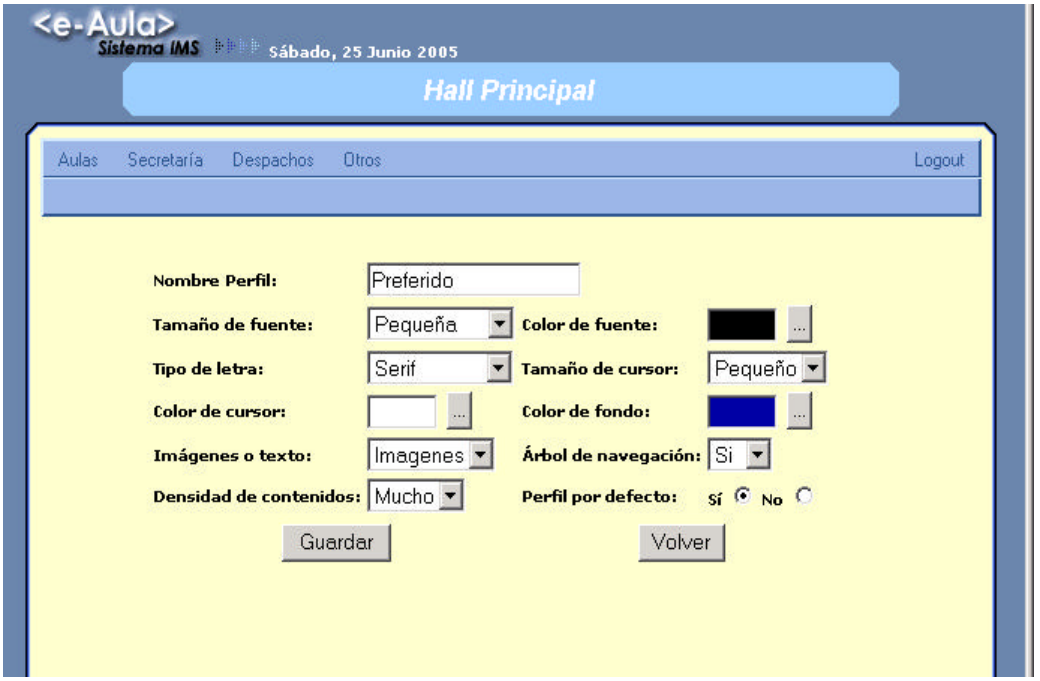

En la esquina superior izquierda tenemos todos los alumnos matriculados en el curso, podemos seleccionarlos como destinatarios del correo pulsando sobre el botón “>>”. También podemos escribir las direcciones a mano en el campo para. O bien seleccionar una de las opciones inferiores: Usuarios activos, activos desde una fecha o matriculados desde una fecha. Al seleccionar una de las opciones que implican fechas nos aparece un calendario donde seleccionar el día.

## 6.- Edición de preferencias del alumno.

El alumno se dirige a la sección de edición de preferencias. Ahí se muestra la lista de los perfiles que hay en el sistema definidos para el usuario.




Como se puede ver en la imagen, desde esta ventana podemos editar o borrar perfiles existentes o bien añadir uno nuevo. Tanto para editar como para añadir un perfil se muestra la ventana con los diferentes valores para los campos que componen el perfil.



# Sección IX. Apéndice c.

# 1.- Ejemplo de un XML descriptivo de un alumno

A continuación se muestra un ejemplo de documento XML que sigue el estándar LIP-ACCLIP de IMS.

Este tipo de documento es el que genera el sistema cuando exportamos el perfil de un alumno.

La parte cursiva indica la localización del sistema, deberá ser sustituida por la URL del sistema

```
<learnerinformation XMLns="http://www.eaula.es/XSD/ims_lip_rootv1p0"
        XMLns:imsacclip=" http://www.eaula.es/XSD/accessforall_v1p0 "
        XMLns:xsi="http:///XMLSchema-instance"
        xsi:schemaLocation="http://www.eaula.es/XSD/ims_lip_rootv1p0
                            http://www.eaula.es/XSD/ims_lip_rootv1p0.xsd
                            http://www.eaula.es/XSD/accessforall_v1p0
                            http://www.eaula.es/XSD/accessforall_v1p0.xsd">
  <identification>
    <formname>
     <typename>
      <tysource sourcetype="imsdefault"/>
      <tyvalue>Preferred</tyvalue>
     </typename>
     <text>Roberto Labreda Sevilla </text>
    </formname>
    <name>
      <typename>
          <tysource sourcetype="imsdefault"/>
          <tyvalue>Preferred</tyvalue>
      </typename>
      <partname>
          <typename>
              <tysource sourcetype="imsdefault"/>
              <tyvalue>First</tyvalue>
          </typename>
          <text>Roberto</text>
      </partname>
      <partname>
          <typename>
              <tysource sourcetype="imsdefault"/>
              <tyvalue>Last</tyvalue>
          </typename>
          <text>Labreda Sevilla</text>
      </partname>
```

```
    </name>
    <contactinfo>
      <email>rolabreda@yahoo.es</email>
    </contactinfo>
</identification>
<activity>
    <typename>
      <tysource sourcetype="imsdefault"/>
      <tyvalue>Education</tyvalue>
    </typename>
    <evaluation>
      <evaluationid>Ejercicio1</evaluationid>
          <result>
              <score>
                <fieldlabel>
                    <typename>
                        <tyvalue>Total</tyvalue>
                    </typename>
                </fieldlabel>
                <fielddata>3</fielddata>
              </score>
          </result>
    </evaluation>
    <description>
      <short>SCORM</short>
    </description>
</activity>
<activity>
    <typename>
      <tysource sourcetype="imsdefault"/>
      <tyvalue>Education</tyvalue>
    </typename>
    <evaluation>
      <evaluationid>Ejercicio 2</evaluationid>
      <result>
          <score>
              <fieldlabel>
                <typename>
                    <tyvalue>Total</tyvalue>
                </typename>
              </fieldlabel>
              <fielddata>8</fielddata>
          </score>
      </result>
    </evaluation>
    <description>
      <short>UML</short>
    </description>
</activity>
<accessibility>
      <imsacclip:accessForAll>
          <imsacclip:context identifier="casa">
              <imsacclip:display>
                <imsacclip:screenEnhance>
                    <imsacclip:screenEnhanceGeneric>
```

```
                    <imsacclip:fontFace>
                        <imsacclip:fontName usage="preferred" value="String"/>
                        <imsacclip:genericFace usage="preferred" value="sansSerif"/>
                    </imsacclip:fontFace>
                    <imsacclip:fontSize usage="required" value="7"/>
                    <imsacclip:foregroundColor usage="required" value="ff000000"/>
                    <imsacclip:backgroundColor usage="required" value="ff0000ff"/>
                    <imsacclip:highlightColor/>
                    <imsacclip:invertedColorChoice/>
                    <imsacclip:cursorSize usage="preferred" value="0.5"/>
                    <imsacclip:cursorColor usage="preferred" value="ff000000"/>
                    <imsacclip:cursorTrails/>
                </imsacclip:screenEnhanceGeneric>
            </imsacclip:screenEnhance>
            <imsacclip:structuralPresentation>
                <imsacclip:contentDensity usage="preferred" value="overview"/>
                <imsacclip:contentViews usage="preferred" value="imageIntensive"/>
            </imsacclip:structuralPresentation>
        </imsacclip:display>
        <imsacclip:control>
            <imsacclip:structuralNavigation>
                <imsacclip:useTableOfContents usage="preferred" value="true"/>
            </imsacclip:structuralNavigation>
        </imsacclip:control>
    </imsacclip:context>
</imsacclip:accessForAll>
</accessibility>
</learnerinformation>
```

## 2.- Scripts de creación de las nuevas tablas del sistema.

```
CREATE TABLE Preferencias
        (User CHAR(20) NOT NULL,
        NombrePerfil CHAR(50) NOT NULL,
        PorDefecto bool NOT NULL,
        FontName CHAR(50) NOT NULL DEFAULT "sansSerif",
         FontSize SmallInt NOT NULL DEFAULT 12,
        BackgroundColor  CHAR(50) NOT NULL DEFAULT "#ff0000",
        CursorSize Float NOT NULL DEFAULT 0.5,
        CursorColor CHAR(50) NOT NULL DEFAULT "#ffffff",
        ForegroundColor CHAR(50) NOT NULL DEFAULT "#ff0000",
        ContentDensity smallint NOT NULL DEFAULT 0,
        ContentViews smallint NOT NULL DEFAULT 0,
        structuralNavigation BOOL NOT NULL DEFAULT 1,
    PRIMARY KEY(User,NombrePerfil),
    FOREIGN KEY(User) references Usuarios(Usuario))
ENGINE=InnoDB;


CREATE TABLE FormatImg(
  id INT NOT NULL auto_increment,
  ext CHAR(20),
  PRIMARY KEY(id)
) ENGINE=InnoDB;

CREATE TABLE TipoImagen(
  tipoImg INT NOT NULL,
  format INT NOT NULL,
  PRIMARY KEY (tipoImg, format),
  FOREIGN KEY(format) references FormatImg(id)
) ENGINE=InnoDB;

CREATE TABLE FormatSound(
  id INT NOT NULL auto_increment,
  ext CHAR(20),
  PRIMARY KEY(id)
) ENGINE=InnoDB;

CREATE TABLE TipoSound(
  tipoSound INT NOT NULL,
  format INT NOT NULL,
  PRIMARY KEY (tipoSound, format),
  FOREIGN KEY(format) references FormatSound(id)
) ENGINE=InnoDB;
```

```
CREATE TABLE FormatVideo(
  id INT NOT NULL auto_increment,
  ext CHAR(20),
  PRIMARY KEY(id)
) ENGINE=InnoDB;

CREATE TABLE TipoVideo(
  tipoVideo INT NOT NULL,
  format INT NOT NULL,
  PRIMARY KEY (tipoVideo, format),
  FOREIGN KEY(format) references FormatVideo(id)
) ENGINE=InnoDB;

CREATE TABLE  DeviceCaract
(
  idDevice CHAR(50) NOT NULL,
  anchoDisplay INT NOT NULL,
  altoDisplay INT NOT NULL,
  tipoImg INT NOT NULL,
  tipoSound INT NOT NULL,
  tipoVideo INT NOT NULL,
  tamCache INT NOT NULL,
  PRIMARY KEY(idDevice),
  FOREIGN KEY(tipoImg) references TipoImagen(tipoImg),
  FOREIGN KEY(tipoSound) references TipoSound(tipoSound),
  FOREIGN KEY(tipoVideo) references TipoVideo(tipoVideo)
)  ENGINE=InnoDB;



CREATE TABLE SubUA(
  subUA CHAR(100) NOT NULL,
  idDevice CHAR(50) NOT NULL,
  PRIMARY KEY(subUA),
  FOREIGN KEY(idDevice) references DeviceCaract(idDevice)
) ENGINE=InnoDB;



CREATE TABLE palPDA(
  pal CHAR(100) NOT NULL,
  PRIMARY KEY(pal)
) ENGINE=InnoDB;
```

```
CREATE TABLE Tiempo(Alumno CHAR(20) NOT NULL,
                    Curso CHAR(50) NOT NULL,
                    Item CHAR(50) NOT NULL,
                    Org  CHAR(50) NOT NULL,
                    Tiempo SmallInt NOT NULL,
                    Fecha DATE NOT NULL DEFAULT 0,
                    IdDispositivo CHAR(50) NOT NULL,
                    Hora TIME NOT NULL DEFAULT 0,
                    Ultima TinyInt(1),
                    PRIMARY KEY(Alumno,Curso,Item, Org),
                    INDEX ind_curso_tiempo(Curso),
                    INDEX ind_alumno_tiempo(Alumno),
                    FOREIGN KEY(Alumno) references Usuarios(Usuario),
                    FOREIGN KEY(Curso) references Cursos(Curso),
                    FOREIGN KEY(IdDispositivo) references
DeviceCaract(IdDevice))  ENGINE=InnoDB;
```

# Sección X. Apéndice d.

# 1.- Bibliografía

ADL (2005). *Advanced Distributed Learning*. http://www.adlnet.org

ADL SCORM v. 2004 (2004). *Sharable Content Object Reference Model*. http://www.adlnet.org/scorm/history/2004/documents.cfm

APACHE. *Struts*. http://struts.apache.org/

CESGA (2005). *Centro de Supercomputación de Galicia.* http://www.cesga.es

CETA (2005). *Centro de tecnologías para el aprendizaje.* Universidad de Santiago de Compostela. http://ceta.usc.es/

EIFEL (2005). *Instituto europeo para el e-learning.* http://www.eife-l.org/eifel

FERNÁNDEZ-MANJÓN, Baltasar., FERNÁNDEZ-VALMAYOR, Alfredo., NAVARRO, Antonio., SIERRA, José Luis. Application of XML Mark-up Languages to Software Development. *Upgrade* (ISSN: 1684-5285), simultáneamente se publica en *Novática* (ISSN: 0211-2124) Volumen: III, nº 4 Páginas, inicial: 15 final: 20 Fecha: Agosto, 2002

FERNÁNDEZ-MANJÓN, Baltasar., SANCHO Pilar. Creating Cost-effective Adaptative Educational Hypermedia Based on Markup Technologies and E-Learning Standards. *Interactive Educational Multimedia*, number 4 (April 2002), pp. 1-11

GOODWILL, James., HIGHTOWER, Richard (2003): *Professional Jakarta Struts*.

IMS (2005). *IMS Global Learning Consortium*. http://www.imsglobal.org

IMS Accessibility (2003). *IMS Accessibility*. http://www.imsglobal.org/accessibility/index.HTML

IMS LIP (2001). *IMS Learner Information Package*. http://www.imsglobal.org/profiles/index.cfm

LÓPEZ MORATALLA, Javier., MARTÍNEZ ORTIZ, Iván., MORENO GER, Pablo (2003). *<e-Aula> : Desarrollo de un sistema e-learning basado en estándares IMS*

MANERO IGLESIAS, Borja (2003). *Estudio de la propuesta IMS de estandarización de enseñanza asistida por computadora*.

SIDAR (2005). *Fundación Sidar.* http://www.sidar.org

UNIVERSIDAD DE TORONTO. *Web-4-All Administrator/User Manual*

UNIVERSIDAD DE TORONTO (2005). *http://www.utoronto.ca/atrc/research/Web4All*

WURLF (2005). *Wireless Universal Resource File.* http://wurfl.sourceforge.net/

XML (2005), *eXtensible Markup Language.* http://www.XML.org

XSL (2005). *eXtensible Stylesheet Language.* http://www.w3c.org/style/XSL

ZYTRAX (2005). http://www.zytrax.com/tech/web/mobile_ids.HTML

## 2.- Palabras clave

- ADL-SCORM 2004
- IMS - LIP - ACCLIP
- J2EE
- Jakarta Struts
- XML
- e-learning
- Aula virtual
- Web accesible
- E-Aula

## 3.- Autorización.

Los autores del presente proyecto autorizan a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado.

Sara Cantos Galán Roberto Álvarez Sánchez José Luis Parrilla Bejerano