UNIVERSIDAD CARLOS III DE MADRID

Escuela Politécnica Superior

Ingeniería de Telecomunicación

Proyecto Fin de Carrera

Implementación de una herramienta de autor generadora de IMS-LD para diferentes pedagogías

Autor: **Francisco José Murcia Sánchez**

Tutor: **Pedro José Muñoz Merino**

Leganés, Junio de 2011

Título: Implementación de una herramienta de autor generadora de IMS-LD para diferentes pedagogías

Autor: Francisco José Murcia Sánchez

Tutor: Pedro Muñoz Merino

EL TRIBUNAL

Presidente:

Secretario:

Vocal:

Realizado el acto de defensa del Proyecto Fin de Carrera el día 05 de Julio de 2011 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de:

Fdo: Presidente Fdo: Secretario Fdo: Vocal

Agradecimientos

La implementación de esta herramienta no hubiera sido posible sin el desarrollo de la especificación IMS-LD por parte del *IMS Global Learning Consortium.* Los documentos de *binding* a XML, *information model* y las buenas prácticas de uso publicadas por *IMS Global* son la base de todo este proyecto.

De no ser por los *frameworks* utilizados, la implementación hubiera resultado mucho más complicada por lo que muchas gracias a Craig McClanahan y *Apache Foundation* (Struts), a Gavin King y *JBoss Community* (Hibernate) y Rod Johnson y su equipo (Spring Framework) y por supuesto, a la comunidad de desarrolladores que contribuye a que existan formas de trabajo cada vez más sencillas y eficientes.

Inestimable es la ayuda que han proporcionado para el desarrollo de este proyecto todas esas personas anónimas que con su participación en foros de discusión planteando y respondiendo dudas han ayudado a superar muchos de los problemas encontrados a lo largo del desarrollo técnico del proyecto.

Nada de esto hubiera sido posible sin la multitud de herramientas utilizadas, todas ellas software libre. En cuanto a herramientas de desarrollo mi gratitud es al grupo de desarrollo de Eclipse (*The Eclipse Foundation*) y a los desarrolladores del paquete de plugins JBoss Tools (*JBoss Community*). Para el despliegue de la aplicación, mi agradecimiento es para el *Apache Software Foundation* por el desarrollo del contenedor de servlets Tomcat y para el grupo de *MySQL AB* por proporcionar un motor de datos ligero, fiable y práctico.

También quiero agradecer al laboratorio *Gradient* de la Universidad Carlos III por el desarrollo de la máquina virtual de *dotLRN* con el player de IMS-LD GRAIL así como a la OUNL por el desarrollo del player de IMS-LD CopperCore. Ambas herramientas han sido fundamentales durante la fase de pruebas de este proyecto.

Y por supuesto, al tutor de este proyecto Pedro Muñoz que ha participado activamente en el diseño y mapeo de las pedagogías y en otras cuestiones relacionadas con IMS-LD y orientación tecnológica. Pedro, muchas gracias por tu dedicación, esfuerzo y tiempo.

Resumen

La especificación IMS-LD proporciona un marco que permite describir un gran número de escenarios pedagógicos. Para ello hace uso de la metáfora del teatro en la que los protagonistas del proceso de formación van asumiendo roles a medida que se ejecutan los actos de la obra. Sin embargo, la especificación no realiza una taxonomía de todas las pedagogías existentes y su manera de realizarse en IMS-LD sino que proporciona un lenguaje flexible para su descripción.

Existen diferentes herramientas de autor capaces de generar ficheros XML finales compatibles con la especificación IMS-LD. Algunas de estas herramientas son muy completas desde el punto de vista de la especificación y están basadas en términos próximos a esta, lo que las hace difíciles de utilizar por los creadores de cursos habituales. Otras de estas herramientas son más cercanas a los términos que habitualmente utilizan los profesores pero sin embargo no son tan completas y sólo abarcan modelos pedagógicos específicos.

En este proyecto fin de carrera se ha implementado una herramienta de autor capaz de generar cursos compatibles con la especificación IMS-LD para un conjunto seleccionado de pedagogías. De este modo, un docente podrá utilizar la herramienta para crear cursos con una orientación pedagógica y generar un contenido IMS-LD válido para ser desplegado en una plataforma de educación online, bordeando las complejidades de la especificación.

Abstract

The IMS-LD specification aims to provide a generic and flexible language to describe the largest number of possible educational scenarios. To do so, uses the metaphor of the theater, in which the protagonists of the training process are assuming roles as they perform the acts of the play. However, although the start point of the specification is a simple model, coding and structure in XML results in a learning curve too hard for teachers and professors.

There are several tools available to create IMS-LD courses, some are powerful software tools which even cover the full specification, but all of them have interfaces based in terms of IMS-LD and this fact drives away potential new users of the specification. Other tools are closer to the vocabulary used by teachers but do not provide full coverage of the specification.

This bachelor thesis is about implementing an authoring tool for generating IMS-LD based courses with an interface where it is possible to choose between different pedagogies. Thus, a teacher can use the tool to create courses with a certain pedagogical approach to be deployed in an online learning platform and avoiding the complexities of the specification.

Definiciones y Siglas

A lo largo de este documento se utilizarán las siguientes siglas y definiciones:

Herramienta de autor	Del inglés "Authoring Tool" o authorware. Se agrupan bajo esta definición aquellas herramientas informáticas que ayudan al usuario a generar hipertexto o aplicaciones multimedia y otros recursos ya sean contenidos, evaluaciones, etc… Típicamente, una herramienta de autor permite al usuario crear una aplicación final simplemente mediante la unión y fusión de elementos.
.LNR	Es una comunidad de educadores, diseñadores y desarrolladores para la innovación en entornos educativos. En lo relacionado con este proyecto se refiere a una plataforma donde desplegar las unidades de aprendizaje.
Coppercore	Es un reproductor de unidades de aprendizaje descritas en IMS-LD
GRAIL	Es un entorno de ejecución de IMS-LD implementado como parte de .LRN.
IMS-LD	IMS Learning Design. Es un metalenguaje para el modelado de escenarios pedagógicos basado en la metáfora de que el proceso educativo puede modelarse como una obra de teatro.
ITS	Intelligent Tutoring System. Es un sistema que proporciona instrucciones o retroalimentación directa a los alumnos sin la necesidad de interacción humana.
J2EE	Java Platform Enterprise Edition. Se trata de una versión de Java para desarrollo de aplicaciones web.
LMS	Del inglés: Learning Management System. Típicamente es una aplicación web que sirve para administrar, distribuir y controlar las actividades de formación no presencial de una institución u organización.
PedaLea	Pedagogical Learning Design. Es el nombre de la herramienta de autor desarrollada para este proyecto.
RTE	Del inglés: Runtime Environment. Son las plataformas en las que las UoL se despliegan para su ejecución.
SCORM	Sharable Content Object Reference Model. Es una especificación que permite crear contenidos pedagógicos estructurados y portables entre plataformas.
UoL	Unit of learning. Se trata de una pieza de aprendizaje. Puede ser desde una lección a un curso entero.
XML	Extensible Markup Language. Es un lenguaje descriptivo de etiquetas consistente en un conjunto de módulos útil para estructurar, almacenar e intercambiar información.

INDICE GENERAL

PÁGINA

APÉNDICES

PÁGINA

FIGURAS

PÁGINA

1. INTRODUCCIÓN

El *e-learning* se puede definir como todo proceso de enseñanza/aprendizaje que emplea medios electrónicos y que normalmente utiliza Internet como entorno de ejecución. La idea de que el *e-learning* aporta interesantes ventajas a empresas, instituciones y usuarios está comúnmente aceptada en la actualidad. La utilización de recursos *e-learning* para la formación de los empleados de una empresa reduce ampliamente los costes en formación tradicional. Para los centros formativos, el *e-learning* permite ampliar y mejorar su oferta docente para llegar, entre otros, a trabajadores que no pueden acceder a sus cursos presenciales, así como un complemento a la docencia presencial [1]. Para los usuarios, el *e-learning* permite el acceso en cualquier momento y lugar a los contenidos docentes y la superación de barreras geográficas, sin olvidar la posibilidad de personalización del aprendizaje y la accesibilidad de los contenidos para usuarios con discapacidades [2].

El sistema educacional de *e-learning* viene desarrollándose desde hace más de una década [3] y durante este tiempo ha sido casi imparable la emergencia de nuevos estándares, plataformas y desarrollos para dar respuesta a una demanda cada vez más creciente de educación a distancia.

El empleo de *e-learning* permite lograr la personalización y la monitorización de la evolución de cada alumno [4], al mismo tiempo que facilita la interacción y el intercambio sencillo de recursos. Si bien es cierto que todas estas últimas facilidades se pueden conseguir sin *e-learning*, su uso ahorra tiempo y reduce los costes económicos ya que por un lado se automatiza el flujo de trabajo de un curso y por otro lado los recursos docentes pueden ser reutilizados.

El aprendizaje con autonomía e independencia da posibilidades de una educación sin la presencia física del docente con las ventajas asociadas en términos de flexibilidad temporal y de ubicación. Las plataformas de *e-learning* proveen de los recursos necesarios para que docentes y alumnos establezcan el necesario canal de comunicación, así como el acceso a los materiales y aplicaciones educacionales basadas en la tecnología. Las principales plataformas de *e-learning* son los LMS (*Learning Management Systems*) y los ITS (*Intelligent Tutoring Systems*).

Un LMS es un paquete de software que permite administrar y proporcionar recursos docentes para los alumnos. La mayoría de los LMS están basados en arquitecturas web para que alumnos y profesores puedan acceder a los recursos en cualquier momento y desde cualquier lugar.

La multiplicidad de plataformas LMS sugiere la necesidad de que existan especificaciones de *e-learning* compatibles con todas ellas. Actualmente existen distintas especificaciones que pretenden llegar a la estandarización tales como SCORM [5], DublinCore [6] o la que centrará la atención de este proyecto: IMS Learning Design [7].

IMS Global Learning Consortium Inc. publicó en 2003 la especificación IMS Learning Design (en adelante IMS-LD). Se trata de una especificación de *e-learning* que propone una manera flexible de modelar escenarios de aprendizaje. Es una forma de crear planes de lecciones para que estos puedan ser interpretados por una aplicación llamada *player* o *run-time environment* (en adelante RET). IMS-LD no ofrece modelos pedagógicos particulares sino que fue diseñado para cubrir un amplio espectro de

escenarios de aprendizaje. IMS-LD es por lo tanto un marco genérico que admite multitud de pedagogías.

1.1. MOTIVACIÓN

Las plataformas LMS son capaces de ejecutar ficheros XML escritos según IMS-LD. Sin embargo los creadores de cursos no tienen porqué entender los detalles técnicos de esta especificación. Es necesario abstraer a los profesores de esa complejidad de modo que se les proporcione la posibilidad de generar cursos de *e-learning* a partir de términos conocidos para ellos.

Para que los profesores puedan bordear el lenguaje tecnológico es necesario proporcionarles una herramienta basada en vocabulario pedagógico. Esta herramienta debería realizar el mapeo a IMS-LD de los parámetros introducidos por los profesores. Las ventajas que aporta IMS-LD a la planificación y a la monitorización de un curso no podrán ser disfrutadas mientras no exista una traducción de la especificación a términos que los profesores puedan entender y manejar con solvencia.

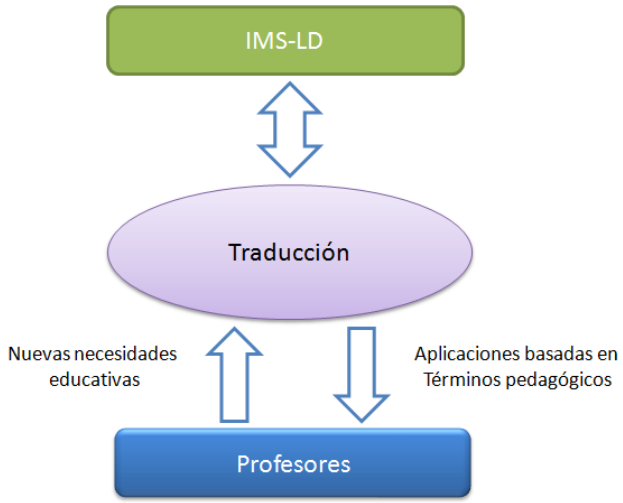


Figura 1-1. Relación existente entre profesores y la especificación IMS-LD

Por un lado tiene que existir una traducción desde el profesorado a los desarrolladores para que estos incorporen las modificaciones necesarias para que se cubra la mayor cantidad de escenarios pedagógicos posibles y se incorporen nuevas necesidades educativas. Por otro lado, los técnicos deben traducir la especificación en un conjunto de conceptos pedagógicos que los profesores puedan manejar con facilidad.

Este proyecto consiste en la implementación de una herramienta que por un lado domine la especificación IMS-LD y por otro lado suministre a los profesores una interfaz basada en términos pedagógicos para crear cursos. La herramienta recibirá el nombre de **PedaLea**, del inglés: ***Peda****gogical* ***Lea****rning Design.*

1.2. OBJETIVOS

Algunos estudios como el realizado por Neumann y Oberhuemer en 2008 [8] revelan que en general, con las herramientas de autor existentes los docentes son capaces de construir con éxito diseños de aprendizaje pero que sin embargo les resulta complicado mapear los conceptos de su entorno de enseñanza a conceptos de IMS-LD.

El objetivo de este proyecto es proporcionar una herramienta software que sirva como puente de unión entre la especificación IMS-LD y los docentes para el desarrollo de algunas pedagogías concretas de forma que la complejidad en el mapeo de términos pedagógicos a términos IMS-LD sea cubierta por la herramienta.

De igual modo, esta herramienta se diseñará de manera flexible para permitir la incorporación futura de nuevas pedagogías o metodologías de aprendizaje.

Con el fin de cumplir los objetivos previstos para esta *"Herramienta de Autor generadora de IMS LD para diferentes pedagogías"* se consideraron de necesario cumplimiento los siguientes requisitos:

- La herramienta será desarrollada en un lenguaje ampliamente conocido (Java, J2EE).
- Para el desarrollo se utilizará un patrón de diseño Modelo Vista Controlador y los *frameworks* de J2EE más conocidos ,como Tiles, Struts, Spring e Hibernate.
- Será una aplicación web para que los usuarios puedan acceder a través de una interfaz conocida como es un explorador web. Además, el hecho de que la aplicación resida en un servidor aislará al usuario de los problemas relacionados de la instalación o del entorno de ejecución, lo que supone una ventaja con respecto al uso de clientes pesados (aplicación de escritorio). Otra ventaja es que esta arquitectura permitirá que la aplicación sea accesible tanto dentro de la intranet de una organización o bien directamente *worldwide*.
- La herramienta será capaz de recoger un conjunto de pedagogías bien conocidas en la literatura y ponerlas en términos entendibles por los profesores.
- El manejo de la herramienta será lo suficientemente sencillo como para que cualquier docente ajeno a la comunidad de las tecnologías de la información pueda hacer uso de la misma. Para ello, se utilizarán controles de formulario estándar.
- La herramienta ocultará al usuario la complejidad del estándar IMS LD. En cambio mostrará expresiones basadas en conceptos pedagógicos.
- Se perseguirá la portabilidad de la herramienta, de manera que el código IMS LD generado sea entendible por todos los RTEs compatibles.
- La herramienta de autor mantendrá un repositorio de componentes que contendrá las UoL previamente generadas por los profesores que usan la herramienta.
- La herramienta de autor será desarrollada bajo el criterio de ser lo más flexible dentro de las posibilidades. De este modo, desarrolladores futuros podrán introducir nuevas pedagogías en la herramienta sin necesidad de efectuar grandes cambios.

- El idioma utilizado en la capa de presentación de la herramienta de autor será tanto inglés como español. De esta manera se pretende conseguir la mayor difusión posible de este proyecto.

1.3. ESTRUCTURA DE LA MEMORIA

En los siguientes capítulos se exponen los detalles en las distintas etapas de estudio y desarrollo de esta herramienta de autor generadora de IMS-LD.

La memoria comienza con la exposición del estado del arte de la relación entre pedagogías e IMS-LD. Para ello, primeramente se expone un análisis teórico de las pedagogías del aprendizaje en el capítulo 2.1 (pág. 5). El apartado 2.2 (pág. 18) recoge una explicación sobre la especificación IMS-LD, su estructura y un extracto de los elementos más representativos que la componen. En el apartado 2.3 (pág. 22) se muestra una compilación de las actuales herramientas de autor para IMS-LD así como los reproductores existentes (RET) en la actualidad. En este capítulo también se recogen las limitaciones de IMS-LD (pág. 25) en cuanto a expresividad pedagógica y la actualidad de la relación entre IMS-LD y pedagogías (pág. 26). Este capítulo concluye con la explicación de la contribución de este proyecto y las cuestiones que quedan abiertas a su conclusión (pág. 33).

En el capítulo 3 (pág. 35) se explica el mapeo a IMS-LD de las pedagogías seleccionadas. En este capítulo se encuentran las consideraciones tenidas en cuenta para el mapeo y los elementos de IMS-LD utilizados para la implementación de cada una de las pedagogías.

El capítulo 4 (pág. 54) abarca las cuestiones relacionadas con los aspectos técnicos de la creación de la herramienta: análisis, diseño e implementación. Se explica por un lado el modelo de datos (pág. 57), la arquitectura de software de la aplicación (pág. 60) y los detalles sobre la librería de IMS-LD que realiza en *binding* a XML (pág. 67). En el capítulo 4.3.6 (pág. 77) se exponen algunos detalles técnicos sobre la integración de los *frameworks* Struts, Spring e Hibernate.

En el capítulo 5 (pág. 81) se recoge un extracto de las pruebas realizadas con la herramienta. Estas pruebas pertenecen por un lado al testeo de la interfaz web y la aplicación en sí misma y por otro lado al testeo de las UoLs generadas por la herramienta al cargarlas en los RETs Coppercore y GRAIL.

En el capítulo de conclusiones y trabajos futuros (pág. 97) se muestran los resultados obtenidos en relación a los objetivo fijados inicialmente, los problemas encontrados durante el desarrollo y la manera de solventarlos. También se expone un resumen de las limitaciones actuales IMS-LD y las posibles mejoras en la especificación, herramientas de autor y RETs actuales.

En el Apéndice B (pág. 112) se encuentra el manual de usuario de la aplicación donde se han incluido las pantallas de la aplicación desarrollada y una guía para el manejo de la misma.

2. ESTADO DEL ARTE

2.1. ANÁLISIS DE LAS PEDAGOGÍAS DEL APRENDIZAJE

El enfoque que se ha querido dar a este análisis no está orientado hacia una extensa explicación psicológica del proceso de aprendizaje sino más bien a la planificación del curso que supone cada pedagogía.

Como base teórica del aprendizaje se puede partir del texto *Aprendizaje y desarrollo humano desde un enfoque sociocultural* [14] donde se identifican los participantes del proceso formativo. Estos participantes forman lo que se conoce como *triada formativa o triángulo formativo* y son: el sujeto en formación (A), el sujeto que enseña (B) y los contenidos procedentes de diversas fuentes de información (C). Se puede ver en la siguiente figura:

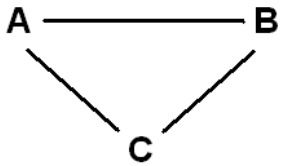


Figura 2-1. El triángulo pedagógico

Cada una de estas entidades A, B y C tiene una misión en el proceso de enseñanza y aprendizaje, dicha misión vendrá descrita por la pedagogía aplicada durante el curso. El éxito en la educación está fuertemente ligado a que la interacción entre el sujeto en formación y el sujeto que enseña sea la adecuada.

Antes de iniciarse un curso escolar, los responsables de la docencia deciden las materias que van a impartir y la metodología que se va a emplear. Por ejemplo, un curso de "Historia de España" se compone, en síntesis, de una serie de unidades didácticas que a su vez contendrán unos contenidos explicativos y unas actividades. La secuenciación o la alternancia entre los distintos elementos que componen un curso, idealmente estará descrita en la planificación previa.

En última instancia, bajo la planificación elegida por el docente, subyace la pedagogía que se está aplicando. Se resumen a continuación las pedagogías más representativas que se fueron desarrollando durante el siglo XX.

2.1.1. El modelo tradicional academicista

El modelo academicista se basa, principalmente, en la presentación y exposición de conceptos por parte del docente pero sin tener en cuenta cómo el alumno adquiere y procesa la información que va recibiendo. Para este modelo enseñar consiste simplemente en exponer, definir y explicar los contenidos de una manera adecuada. De otra manera, se podría decir que el foco de este modelo está centrado en la morfología, la estructura y el contenido del curso, dejando a un lado el aprendizaje de los alumnos.

El hecho de que el profesor no preste atención a la manera de seguir el curso por parte de los alumnos destruye la realimentación que debe existir entre alumnos y profesores, las circunstancias personales de

cada alumno no son tenidas en cuenta y precisamente por eso no exprime toda la potencialidad de un grupo de alumnos.

El éxito del proceso de enseñanza y aprendizaje con este modelo está limitado por las aptitudes y las capacidades individuales de cada alumno para estudiar por su cuenta. Las tareas básicas del alumno durante un curso con modelo academicista son las de asistir a clase, escuchar las diferentes lecciones, tomar apuntes y la memorización de los conceptos

El tipo de planificación que se ajusta a este modelo es el conocido como sábana. Se trata de que el profesor haga una enumeración y una explicación de los conceptos y que, por su parte, los estudiantes los memoricen en un determinado plazo de tiempo.

2.1.2. El conductismo

El filósofo ruso Ivan Pavlov fue quien desarrolló los primeros trabajos que dieron lugar a las teorías conductistas. Pavlov [15] enfatizaba la fisiología y el papel de los estímulos en producir respuestas condicionadas. Es muy conocido el experimento del "perro de Pavlov" aunque fueron varios animales a los que el filósofo ruso sometió a experimentos. Sus estudios pudieron demostrar que un estímulo inicialmente neutral para un sujeto podía convertirse en un estímulo que provocara una respuesta determinada y también determinista en ese mismo sujeto.

Existen tres aproximaciones conductistas, la primera es el condicionamiento clásico que está basado, precisamente, en los experimentos de Pavlov y centra su atención en actos reflejos y fisiológicos de animales. Seguidamente se muestra el condicionamiento instrumental y finalmente el condicionamiento operante.

2.1.2.1. Condicionamiento clásico

Se puede decir que el aprendizaje, como reacción condicionada, es el tipo de aprendizaje más básico. Para que se dé este tipo de aprendizaje es necesario que exista un entrenamiento repetitivo, de tal modo que la respuesta al estímulo condicionado sea constante y casi mecánica.

Los conductistas explican la psicología de los seres humanos como un sistema mecánico que responde a los estímulos a partir de asociaciones previamente establecidas. Los conductistas son capaces de explicar todas las conductas de los seres humanos a partir de su condicionamiento previo.

Basándose en los estudios de Pavlov, J. Watson fue uno de los máximos exponentes del condicionamiento clásico. Fue más allá del estudio de los simples actos reflejos y se centró en el estudio de la conducta humana, concretamente en el control y la predicción del comportamiento.

El condicionamiento clásico en la enseñanza es frecuente, fundamentalmente se emplea para el aprendizaje de automatismos o hábitos de conducta y conocimientos puramente memorísticos. Para los conductistas, el aprendizaje se consigue a través del entrenamiento, para conseguir un refuerzo que la lógica estímulo-respuesta se afiance en el alumno. Por lo tanto, se pretende que el estudiante aprenda a responder siempre lo mismo frente a un mismo estímulo.

2.1.2.2. Condicionamiento instrumental

E. Thorndike también desarrolló las tesis pavlovianas y enunció la *teoría del aprendizaje por ensayo y error*. Además, Thorndike, enunció las conocidas como *leyes del aprendizaje* que describen la interrelación entre estímulos y respuestas. Lo más importante de los trabajos de Thorndike es que da importancia a la motivación en el aprendizaje [18]. Los trabajos de Thorndike se basan en el aprendizaje de una conducta como medio para lograr una recompensa.

La misión del docente en este modelo es la de seguir la planificación previamente elegida del curso o el libro de texto asignado. Además, es importante la repetibilidad durante el proceso de aprendizaje y que esta sea sin fisuras. Es decir, si el alumno espera una recompensa por un trabajo bien hecho, es indispensable proporcionársela para que se refuerce la asociación comportamiento-recompensa.

2.1.2.3. Condicionamiento operante

El condicionamiento clásico presenta ciertas deficiencias derivadas del hecho de que se centra fundamentalmente en la asociación entre estímulos sensoriales y respuestas reflejas. Burrhus Frederic Skinner [24] fue un paso más allá y diferenció entre comportamiento de reacción (Pavlov) y el comportamiento operante. Este último no se limita a reaccionar ante estímulos sino que trata de analizar la relación que existe entre las conductas de los sujetos y sus consecuencias, dicho de otro modo, .se plantea el concepto de refuerzo del aprendizaje debido al éxito y al concepto de retroalimentación.

Para demostrar sus teorías Skinner introdujo a un roedor dentro de una caja que contenía una palanca que le proporcionaba comida. Nada más llegar a la caja el comportamiento del roedor era errático, yendo de un lado para otro. Por casualidad el roedor accionaría la palanca y recibiría una recompensa en forma de alimento. Durante las siguientes repeticiones del experimento, el tiempo errático del roedor se iría reduciendo y la palanca sería accionada cada vez antes. Con este sencillo ejemplo, Skinner demostraría que aquellos comportamientos que producen placer o recompensa serían repetidos frente a aquellos que no lo producen.

La principal aportación de Skinner a la educación ha sido la enseñanza programada. De hecho, llegó a diseñar una "máquina de enseñar" [23] que produjo una gran revolución en la época (mitad del siglo XX). Las bases que sustentaban esta máquina eran el desglose de la materia en pequeños pasos, el refuerzo inmediato y el repaso continuado, sin embargo, dicho artilugio nunca llegó a implantarse.

Al igual que el condicionamiento clásico y el condicionamiento instrumental, el condicionamiento operante es útil para el aprendizaje o la corrección de hábitos o conductas, que se refuerzan o debilitan debido a sus consecuencias.

2.1.3. El cognitivismo

El cognitivismo [19] estima que el aprendizaje es el resultado de una reorganización de percepciones y de la formación de nuevas relaciones. En contraposición al conductivismo, los cognitivistas defienden que el ser humano aprende cuando construye conocimiento sobre otros conocimientos previamente

adquiridos. Para los cognitivistas el aprendizaje es un proceso más complejo que va más allá de la asociación estímulo-respuesta de los conductistas.

Desde el punto de vista de la planificación del curso, en el planteamiento cognitivista es muy importante que el profesor conozca los conceptos que sus alumnos ya dominan. De este modo, el profesor podrá ir construyendo nuevos conceptos en los alumnos, sobre la base que forman aquellos que ya tienen adquiridos. La manera en la que los alumnos crean nuevos conceptos debe ser progresiva, consiguiéndose durante el desarrollo del curso una progresión intelectual de los sujetos en educación. Precisamente por esto, los planes de estudios cognitivistas deben partir de elementos básicos hacia elementos más complejos.

Las teorías cognitivas han desarrollado multitud de modelos, los tres más significativos son el aprendizaje por descubrimiento (Brurner), el aprendizaje significativo (Ausubel) y el aprendizaje constructivista (Novak). A continuación se muestra un resumen de estos modelos.

2.1.3.1. El aprendizaje por descubrimiento

El aprendizaje por descubrimiento, fue identificado en los años 70 por Jerome Bruner. Esta teoría atribuye una gran importancia al aprendizaje por ensayo y error [27]. Mediante este tipo de aprendizaje, se fomenta la capacidad de inducir conocimiento por parte de los alumnos. Bruner, convencido de la correlación positiva entre la participación activa del alumno en clase y la calidad del aprendizaje, centró su preocupación en cómo motivar al aprendiz a una participación activa.

Para implementar este tipo de aprendizaje, los profesores deben proporcionar situaciones problemáticas para que los alumnos aprendan mediante el proceso de búsqueda de la solución. Debido a esta forma de aprendizaje no guiada, el profesor debe cuidar el material aportado para que, en realidad, se alcance el propósito de estimular a los alumnos en la búsqueda y descubrimiento del conocimiento.

Para fomentar el aprendizaje inductivo por parte de los alumnos, el curso se planteará de manera heurística más que de manera expositiva. De esta manera, se fomentará que los alumnos prueben a tantear las posibles soluciones de un problema para dar ellos mismos con la solución. En este tipo de aprendizaje suele ser de gran ayuda para los alumnos la presentación de ejemplos prácticos por parte del profesor, ya que de estos ejemplos son capaces de extraer, no la solución, sino la manera de llegar a ella.

Desde el punto de vista de la planificación, este modelo de aprendizaje se ajustaría a una programación en espiral. Esto quiere decir que la asignatura comenzaría en los conocimientos más básicos para posteriormente retomarlos de nuevo, de manera más compleja.

2.1.3.2. El aprendizaje significativo

Ausubel propone la teoría del aprendizaje significativo [20], en 1973 como alternativa al aprendizaje por descubrimiento. Por un lado, Ausubel apoyaba el aspecto cognitivista embebido en el aprendizaje por descubrimiento. Sin embargo Ausubel opinaba que el aprendizaje por descubrimiento era una forma de

aprendizaje ineficiente ya que un aprendizaje medianamente complejo podría alargarse demasiado en el tiempo [28].

Como miembro de la escuela cognitivista, Ausubel definió que el aprendizaje significativo ocurría cuando un concepto se fusionaba con otro. De esa manera, los conceptos más sencillos se unirían para formar los cimientos de otros conceptos más complicados. Para que esta fusión de conceptos lleve hacia un aprendizaje de calidad es primordial que los conceptos preexistentes en la estructura cognitiva del alumno estén lo suficientemente claros y afianzados.

El aprendizaje significativo es también la base de la Teoría de Educación de Novak, pese a que es Ausubel quien identifica el papel crucial que tiene la predisposición del alumno en el proceso de construcción de significados, es Novak quien le da carácter humanista al término, ya que tiene en cuenta la experiencia emocional en el proceso de aprendizaje [28] y [29].

Ausubel encontró una posibilidad en el hecho de completar el aprendizaje por descubrimiento con el aprendizaje por recepción. Se podría definir, de manera sencilla que el aprendizaje por recepción es aquel que se obtiene de una exposición o una explicación. El aprendizaje significativo considera que la instrucción es el medio más adecuado para que exista aprendizaje significativo. El proceso de instrucción tiene que comenzar con el conocimiento, por parte del instructor, de aquellos conceptos que el alumno ya posee. No sólo es necesario que el profesor conozca la enumeración de conceptos que el alumno conoce sino hasta qué punto es capaz de dominarlos y cómo están de arraigados en la estructura cognitiva del alumno dichos conceptos. De esa manera, el profesor puede plantear el temario de manera que la secuencia de exposiciones pueda generar en el alumno nuevos conocimientos. La utilización de mapas conceptuales ayudará en el proceso de enseñanza-aprendizaje.

Los mapas conceptuales [25] son un concepto evolucionado del esquema tradicional ya que incluyen una presentación gráfica de los contenidos de manera que de un sólo vistazo, el alumno puede situarse dentro del curso. Los mapas conceptuales encajan perfectamente dentro del aprendizaje significativo debido a que sitúan al alumno en la posición adecuada para discriminar entre aquellos conceptos que ya conoce y aquellos nuevos que tendrá que incorporar. Con la utilización de los mapas conceptuales, el alumno es capaz de organizar mejor los nuevos conocimientos que está recibiendo y por lo tanto es capaz de relacionarlos fácilmente con sus conocimientos previos. Precisamente ésta es la base del aprendizaje significativo.

La planificación que más se ajusta a este tipo de aprendizaje es la instrucción. Los contenidos que el profesor aporte a los alumnos deberán ser impartidos de manera secuencial y lo más organizada posible. La utilización de mapas conceptuales será de gran ayuda para conseguir un aprendizaje significativo satisfactorio. Pero lo más importante es el conocimiento por parte del profesor de aquello que ya conocen sus alumnos, ya que según el propio Ausubel: *"[...] el factor más importante que influye en el aprendizaje es lo que el alumno ya sabe. Averígüese esto, y enséñese en consecuencia*".

2.1.3.3. El aprendizaje constructivista

El paradigma constructivista propone que el aprendizaje se consigue mediante la transformación del conocimiento que el alumno posee. En este proceso de transformación, las capacidades intelectuales

del alumno actúan como catalizadoras, permitiendo que el conocimiento antiguo se convierta en conocimiento nuevo. Cuanto mayor sea la implicación activa del alumno y su capacidad de asociación, mejor resultará el proceso de transformación del conocimiento.

Las teorías del aprendizaje constructivista defienden que la docencia debe realizarse mediante acciones y actividades en lugar de hacerse mediante la explicación abstracta de conceptos. El alumno debe sentirse en contacto con aquello que está aprendiendo y de esa manera involucrarse de manera activa y desear aprender. Los constructivistas opinan que mediante la experimentación, el alumno creará unos conocimientos basados en procedimientos de ensayo y error que serán de alta calidad.

Durante el desarrollo de un curso de base constructivista, es fundamental que exista un flujo de comunicación entre alumnos-profesores y también entre alumnos-alumnos. Por un lado, gracias a la comunicación entre alumnos y profesores, los alumnos recibirán el *feedback* que necesitan para poder superar algunas dificultades y además se podrán abrir procesos de consenso y discusión que pueden ser de gran utilidad para el aprendizaje. Por otro lado, la comunicación alumnos-alumnos permite que el individuo se involucre en el grupo para ir construyendo sus propios conceptos, en lo que sería la dimensión social de la construcción del conocimiento.

El aprendizaje constructivista centra su interés en el progreso de los alumnos durante el curso, por eso, este modelo también es adecuado para la programación de cursos en espiral, en los cuales se comienza a impartir contenidos sencillos para posteriormente regresar sobre ellos de una manera más exhaustiva. Con esta dinámica del curso es posible generar conocimiento en los alumnos de una manera progresiva y que al final éstos dispongan de nuevos conceptos claros y útiles para seguir construyendo nuevos conocimientos.

2.1.4. El aprendizaje en grupo. Aprendizaje colaborativo y cooperativo

Se podría definir el aprendizaje en grupo como un "constructivismo social" [30] en el sentido en el que las doctrinas cognitivistas-constructivistas mantienen su vigencia pero añadiendo la dimensión social de los alumnos. Para que suceda el aprendizaje en grupo, el aula debe ser un entorno de exposición de ideas y de debate. La interacción entre los alumnos supone que, al final, cada alumno hace su interpretación personal de los conocimientos y de esa manera construye su conocimiento en base a aquello que previamente conocía [31].

Podrían definirse dos tipos de aprendizaje en grupo: el aprendizaje colaborativo y el aprendizaje cooperativo. A primera vista, colaborar y cooperar parecen sinónimos, pero existen diferencias. Por un lado, el aprendizaje cooperativo supone dividir un problema para que cada alumno resuelva una parte por separado. Por otro lado, el aprendizaje colaborativo supone que los alumnos trabajen conjuntamente para llegar a la solución de un problema. Se puede ver más claro con un ejemplo: supongamos que a un grupo de tres alumnos se les propone un problema de programación. Los tres alumnos estarán cooperando si dividen el problema en tres partes y cada uno resuelve la suya por separado. Sin embargo, los tres alumnos estarán colaborando si los tres se sientan juntos delante de un ordenador a programar todo el código.

Desde el punto de vista de la planificación es interesante observar el siguiente gráfico extraído de [32]:

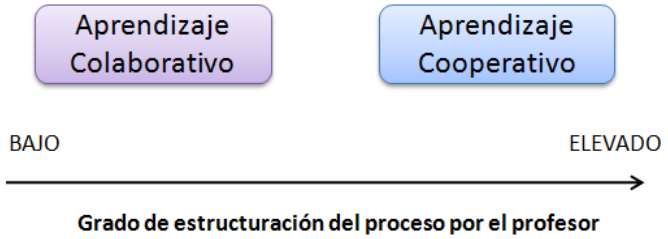


Figura 2-2. Grado de estructuración del proceso de enseñanza-aprendizaje (ref. [32])

El grado de estructuración es elevado en el aprendizaje cooperativo porque es necesario que el profesor realice una división de tareas a repartir entre los alumnos. El aprendizaje colaborativo requiere de un grado de estructuración mucho menor porque en este tipo de aprendizaje lo que se pretende es dejar en manos del grupo de alumnos la responsabilidad del aprendizaje.

En el aprendizaje cooperativo, los alumnos comparten un objetivo común que se alcanzará cuando todos los integrantes del grupo cumplan con su tarea asignada. En el aprendizaje colaborativo la idea es que el conocimiento se adquiera a través de la interacción del grupo de manera que el grupo de alumnos se forma por la fusión de las individualidades de sus integrantes. En este sentido, el aprendizaje colaborativo presenta el riesgo de corromperse y de llevar al grupo hacia la conformidad en los resultados y hacia los retrasos en la realización de las actividades.

Es importante destacar que, en última instancia, el aprendizaje es un proceso que sucede en el individuo. Sin embargo, la posibilidad de aprender en grupo permite al alumno obtener el flujo de realimentación necesario para que exista el aprendizaje.

2.1.5. Aprendizaje basado en problemas

En el aprendizaje basado en problemas el profesor plantea un problema sin haber explicado antes una clase ni otros ejercicios previos. La clave en este tipo de aprendizaje es que el alumno descubrirá el contenido que sea adecuado para resolver el problema.

La variante grupal del aprendizaje basado en problemas plantea que los alumnos deben resolver los problemas en pequeños grupos, para posteriormente debatir las diferentes soluciones, bajo la supervisión del profesor.

Con este tipo de aprendizaje, no sólo se consigue la adquisición de conceptos por parte de los alumnos, sino que también existe un aprendizaje de conductas y actitudes. La habilidad de los alumnos a la hora de enfrentarse a problemas reales mejora considerablemente gracias a este tipo de aprendizaje.

A continuación, se muestran diferentes aproximaciones de aprendizaje basado en problemas.

2.1.5.1. Método de los cuatro pasos de Polya

George Polya nació en Hungría en 1887. Uno de los trabajos más importantes de Polya es el "Método de los cuatro pasos", en el que describe la forma en la que los alumnos deben acercarse a los problemas de carácter matemático para poder resolverlos. Los pasos son los siguientes:

Entender el problema

Configurar un plan

Ejecutar el plan

Mirar hacia atrás

Para que un alumno entienda el problema tiene que plantearse algunas preguntas como por ejemplo si sería capaz de escribir el enunciado con sus propias palabras o detectar si el problema contiene datos extraños.

A la hora de configurar un plan, el alumno tiene que hacer uso de las estrategias de resolución de problemas que conoce para detectar cuál es la que mejor se ajusta al problema planteado. Una de las formas más sencillas de abordar un problema es mediante ensayo y error. Sin embargo, un alumno avanzado puede probar otras técnicas heurísticas para tratar de solucionar el problema.

El siguiente paso es ejecutar el plan decidido y resolver el problema. Finalmente el alumno debe observar el resultado final y plantearse, por un lado, si el resultado al que ha llegado responde a la pregunta que le hacía el problema, y por otro lado evaluar si el resultado es lógico para el problema planteado.

Utilizando la base fundamental del "Método de los cuatro pasos" de Polya, se puede generar un plan de estudios de aprendizaje basado en problemas (APB). A continuación se muestran posibles implementaciones.

2.1.5.2. Universidad de Masstricht, Holanda

La Universidad de Masstricht, tiene una larga experiencia en el aprendizaje basado en problemas. En esta Universidad el curso se divide en módulos relacionados entre ellos que duran desde 6 hasta 12 semanas. El planteamiento en la Universidad de Masstricht es el siguiente: los profesores comienzan por conocer cuáles son los conocimientos que los alumnos tienen previamente adquiridos, esta parte es fundamental para que un curso de aprendizaje basado en problemas sea provechoso. El siguiente paso es plantear a los alumnos el problema y que éstos entiendan lo que se les pide resolver. Para garantizar que el alumno entiende la motivación y el alcance del problema que se pide resolver, el profesor puede iniciar un diálogo en el que los alumnos le plantearán aquellas cuestiones que no tengan claras del problema.

Una vez que el alumno conoce el problema, tratará de resolverlo con las herramientas conceptuales de las que dispone. Comienza de este modo, el estudio individual del problema por parte de los alumnos. En última instancia, su habilidad para relacionar conceptos determinará el éxito a la hora de abordar y resolver el problema planteado. También será muy importante que el profesor oriente al alumno acerca del los pasos que tiene que dar para alcanzar la solución.

Los pasos para la solución de un problema según la Universidad de Maastricht, Holanda son [33]):

1. Clarificación de los términos y conceptos en la descripción del problema.
2. Definición del (los) problema(s)
3. Análisis del problema (lluvia de ideas)
4. Organización de las ideas propuesta en el paso 3
5. Formulación de objetivos de aprendizaje

6. Obtención de nueva información
7. Reporte de los resultados en el grupo tutorial

Gráficamente se muestra el proceso del aprendizaje basado en problemas en la siguiente figura:

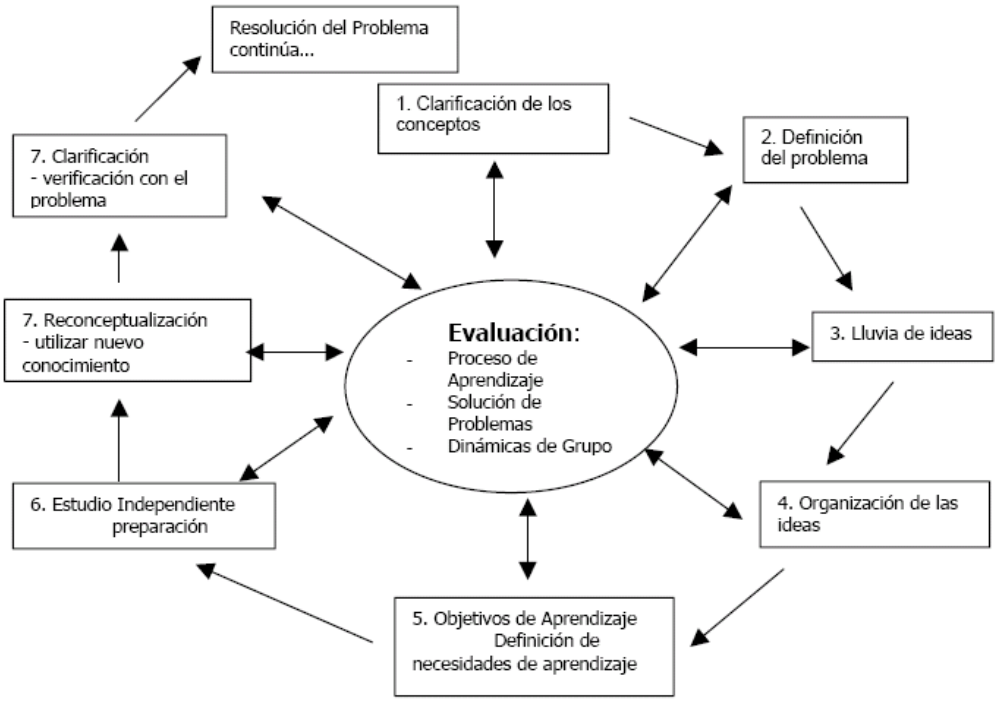


Figura 2-3. Proceso de aprendizaje basado en problemas en la Universidad de Maastrich, Holanda (ref. [33])

Subyace en este tipo de aprendizaje la necesidad de que el alumno muestre una actitud activa para buscar la solución del problema. Será tarea del profesor mantener la motivación del alumno para que esto suceda. El hecho de que los resultados obtenidos sean expuestos y debatidos en la clase, fomenta en los alumnos su pensamiento creativo y su capacidad para desenvolverse en grupo.

Por último, es interesante destacar que para que este tipo de aprendizaje sea fructífero, es necesaria la intervención del profesor como evaluador de los problemas que se plantean. De esta manera el alumno obtiene el feedback que necesita para afianzar los conocimientos y progresar.

2.1.5.3. Universidad Jesuita de Wheeling

Este enfoque del aprendizaje basado en problemas es similar al anterior y está basado en los mismos principios de identificación y resolución de problemas. Como diferencia con el anterior enfoque se puede observar que éste no hace hincapié en la separación en subgrupos para la resolución de problemas.

Esquemáticamente se observa en la siguiente figura cómo se entiende en aprendizaje basado en problemas por la Universidad Jesuita de Wheeling:

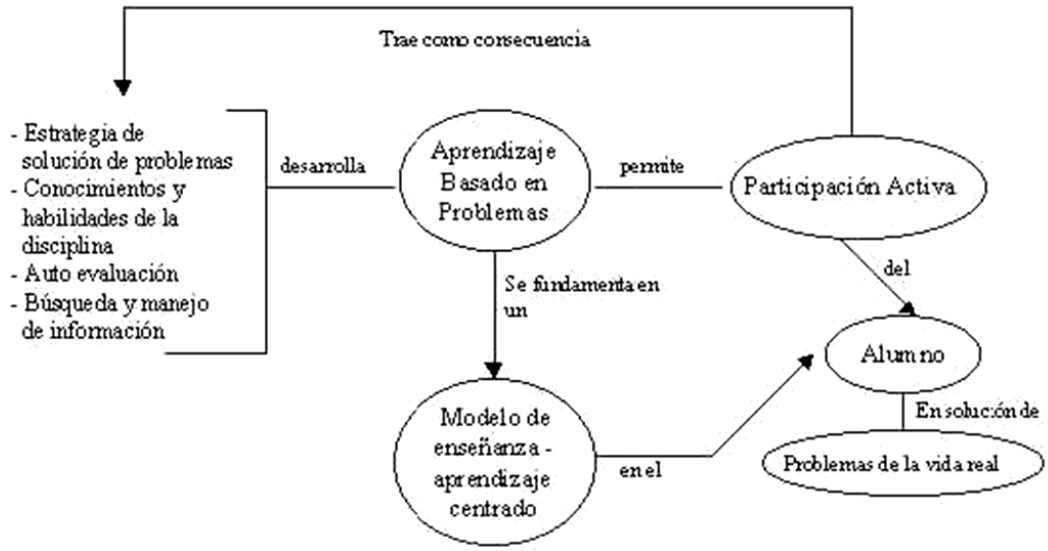


Figura 2-4. Esquema del aprendizaje basado en problemas según la Universidad Jesuita de Wheeling (ref. [33])

Los distintos pasos a la hora de abordar un problema vienen expresados en la siguiente figura:

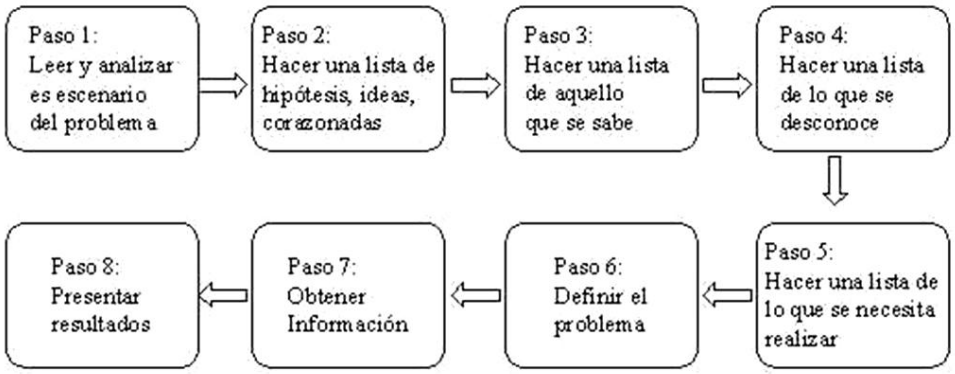


Figura 2-5. Pasos del aprendizaje basado en problemas según la Universidad Jesuita de Wheeling (ref. [33])

Al igual que en la visión de la universidad de Masstricht el alumno debe presentar una actitud activa para que se involucre en la búsqueda de la solución al problema que le han planteado. El alumno debe planificar la investigación que va a realizar y además compartirla y consensuarla con su grupo de trabajo, en el caso de que se plantee la solución de un problema mediante grupos. Por su parte, el profesor debe jugar el papel de asesor de los alumnos en su investigación del problema y proporcionar a los alumnos el *feedback* necesario. También en que el profesor informe, de manera previa, acerca de cómo va a evaluar a los alumnos.

2.1.5.4. Aprendizaje basado en problemas para ingenieros

Esta particularización del aprendizaje basado en problemas ha sido analizada y descrita por un grupo de trabajo de la *Escola de Engenharia de São Carlos* [38] y propone una aplicación del aprendizaje basado en problemas incorporando las particularidades de un curso para estudiantes de ingeniería.

Los autores de esta metodología basan sus argumentos en las teorías cognitivistas de Ausubel que de manera resumida tienen su núcleo en que *"la solución a cualquier problema supone la reorganización de los recuerdos experimentales y la adaptación de los mismos a la situación actual"*.

Los ingenieros trabajan con sistemas resultantes de la unión de otros subsistemas, en ocasiones genéricos y otras ocasiones con particularidades concretas. Los ingenieros por lo tanto deberían integrar su conocimiento y su experiencia en cada nuevo sistema que encuentran a lo largo de su vida profesional intentando identificar las características comunes y así poder priorizar y solucionar los problemas. Mediante el uso de este tipo de aprendizaje, los alumnos incorporan destrezas fundamentales para su desarrollo profesional como por ejemplo, la utilización de los recursos docentes necesarios para resolver un problema.

El proceso de un aprendizaje basado en problemas comienza con la presentación del enunciado del problema por parte del profesor para que posteriormente los alumnos puedan trabajar en grupos para conseguir identificar las tareas a realizar y también cuáles son sus carencias para resolver el problema. Se trata pues de un proceso interactivo que puede resumirse en los siguientes pasos:

1. **Presentar el enunciado.** Se propone en este punto cuál es el problema a resolver. Normalmente será un escenario complejo al que dar respuesta. Por ejemplo, una red de datos que tiene algún problema de eficiencia o un problema completo a resolver como diseñar una infraestructura nueva.
2. **Listar aquello que se conoce.** Los estudiantes se agrupan para tratar de responder a la pregunta: *"¿Qué es lo que sabemos?"*. La respuesta podrá incluir una revisión de los aspectos del enunciado así como un listado de aquello que ya se conoce gracias al conocimiento previo.
3. **Desarrollar un enunciado para el problema.** A partir del enunciado inicial y aquello que los alumnos conocen, se deben redefinir en este punto los diferentes aspectos que no hubieran quedado detallados en el punto 1. De este modo, todos los alumnos parten de un enunciado común que ha surgido de su interpretación del problema y de aquello que ya conocen.
4. **Listar aquello que no se conoce.** En este punto los alumnos deberían responder a la pregunta: *"¿Qué necesitamos saber?"*. Una vez que el escenario ha quedado clarificado, los estudiantes deben identificar qué es lo que necesitan aprender para solucionar el problema.
5. **Listar alternativas o hipótesis.** En este punto se debería responder a la siguiente pregunta: *"¿Qué deberíamos hacer?"*.
6. **Presentar y defender la solución.** Una vez realizados los pasos anteriores, el docente podría requerir de los alumnos una defensa de las soluciones a las que hubieran llegado.

Esta forma de resolver los problemas coloca al alumno como absoluto protagonista del proceso de aprendizaje, siendo el propio alumno el que tiene que descubrir en sí mismo cuáles son sus conocimientos y carencias y buscar posteriormente los materiales que necesita para resolver el problema.

Durante todo el proceso de resolución del problema, los docentes deberían actuar como consultores del problema. Los profesores por lo tanto deberían participar y estimular la participación de los alumnos en esta metodología.

2.1.6. Aprendizaje orientado a proyectos

La idea fundamental de este tipo de aprendizaje es que el alumno aprenda una metodología para la consecución de un proyecto. A diferencia con el aprendizaje basado en problemas, un proyecto es algo mucho más complejo y que requiere que el alumno tenga que planificarse y organizarse de cara a la realización de un trabajo. El aprendizaje orientado a proyectos es útil en las enseñanzas universitarias, ya que preparan al alumno en la dinámica de su futuro trabajo.

Para poder aplicar la estrategia de aprendizaje orientado a proyectos es necesario que el profesor defina primeramente el tipo de modelo a utilizar. Entre otros aspectos, el modelo se define en función de los cursos que abarcará el proyecto. A continuación se presentan dos posibles modelos:

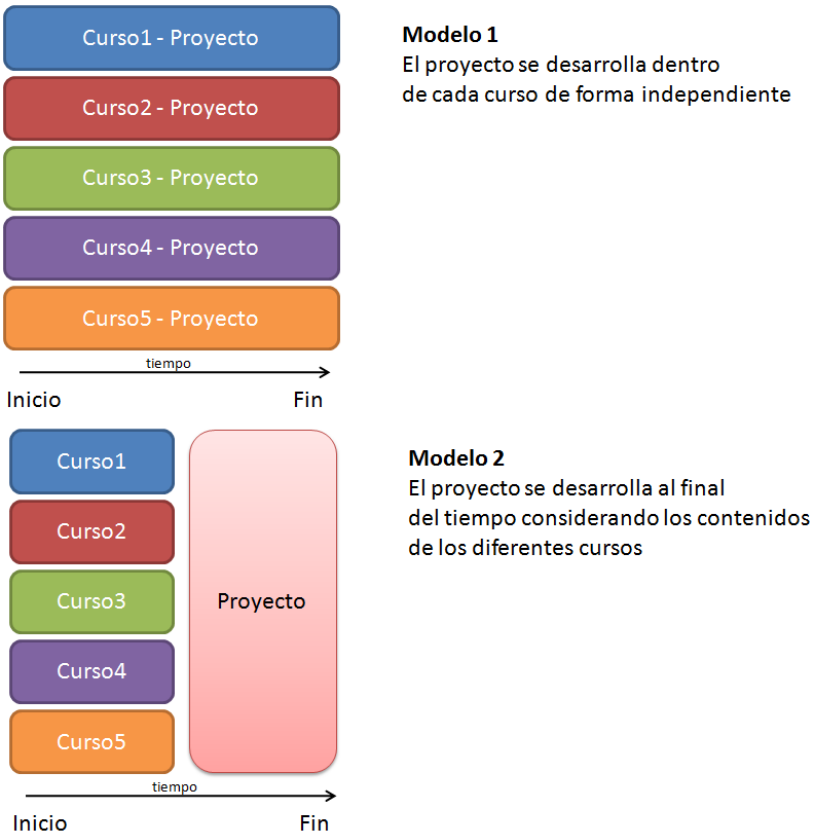


Figura 2-6. Ejemplos de planificaciones para cursos cuyo aprendizaje está basado en proyectos (ref. [34])

A parte de estos dos modelos, otros modelos de realización de proyectos durante el curso son posibles, dependerá de las características de los proyectos. El aprendizaje orientado a proyectos es muy útil para enseñanzas superiores ya que conllevan la preparación de los estudiantes a la dinámica del mundo laboral que, casi siempre, está regido por proyectos. Con este tipo de aprendizaje se fomenta la colaboración entre los alumnos y la capacidad de gestionar su propio tiempo. A continuación se muestran dos ejemplos de aprendizaje basado en proyectos:

2.1.6.1. Proyectos en ciclo de vida

Se trata del caso típico del abordaje de un proyecto en fases de manera que cada fase sea ejecutada por un grupo de trabajo de un perfil concreto. Como por ejemplo el desarrollo de un paquete de software:

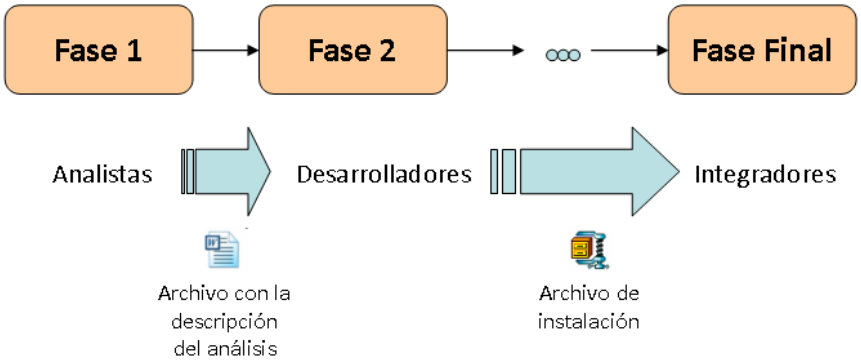


Figura 2-7. Esquema para el aprendizaje basado en proyectos en ciclo de vida

Como se puede ver en la figura, cada fase será ejecutada por un grupo de personas de un rol específico que aportarán a los miembros de la siguiente fase un *output* que será la base de su trabajo y proporcionarán a los miembros de la siguiente fase en *input* necesario.

2.1.6.2. Proyectos en desarrollo simultáneo

En este caso se trata de simular un proyecto o situación multidisciplinar en el que todas las fases son ejecutadas de manera simultánea por diferentes roles. Un ejemplo de este tipo de proyecto sería la simulación de un juicio en una clase:

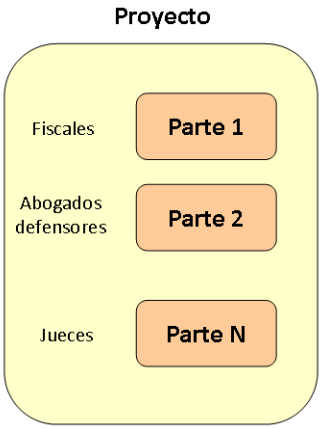


Figura 2-8. Esquema para el aprendizaje basado en proyectos en desarrollo simultáneo

Como se puede observar en la figura anterior, el proyecto es en realidad una situación multidisciplinar en la que cada grupo de trabajo se encargaría de una parte, asumiendo un rol conjunto. El objetivo de este tipo de proyectos sería poder simular en un grupo de trabajo una situación real repartiendo los diferentes roles implicados entre diferentes grupos de alumnos.

2.2. ESPECIFICACIÓN IMS-LD

2.2.1. ¿Qué es IMS-LD?

La especificación IMS-LD [7] fue concebida con el objetivo de codificar escenarios de aprendizaje. Una de las premisas de partida de la especificación es la de ser pedagógicamente neutra, por ello, en IMS-LD no se establecieron parámetros para modelar pedagogías concretas sino que se definió un marco general. En teoría, cualquier pedagogía de las explicadas en el anterior apartado podría ser implementada [39] utilizando IMS-LD.

En última instancia IMS-LD se puede ver de manera sencilla como un archivo XML (*eXtensible Markup Language*) que incorpora una secuenciación de actividades a realizar por los alumnos. De esa manera, al ejecutarse por un intérprete de IMS-LD se recrea la estructura, asignación de roles, sincronización, y la secuenciación del curso.

El concepto central de la especificación IMS-LD es que el aprendizaje es un proceso orientado hacia la consecución de ciertos objetivos docentes en el que el alumno asume un rol y el docente asume otro. Este proceso de aprendizaje debe suceder en un entorno estructurado de alguna manera. Esta definición de aprendizaje es lo suficientemente genérica como para dar cabida a multitud de pedagogías.

Profundizando un poco más, IMS-LD está pensado para crear unidades de aprendizaje, en inglés *units of learning* (UoL). Una UoL se puede definir como un proceso educativo limitado en el tiempo como por ejemplo, un curso completo o una única lección. Es importante tener en cuenta que una UoL no es solamente un conjunto de información que se quiere transmitir al alumno sino que puede incluir actividades, evaluación, recursos, foros de discusión, servicios...

Tanto la secuenciación de los contenidos, como los roles que asumirán profesor y alumno estarán definidos por el *learning design* descrito para la UoL.

2.2.2. Estructura de IMS-LD

La especificación de IMS-LD se reparte en diversos documentos tales como un modelo conceptual y un modelo de información. En ellos se definen y especifican los conceptos básicos y relaciones dentro de IMS-LD además de una descripción de los elementos y atributos XML que pueden ser definidos.

En IMS-LD se han definido tres niveles de implementación: A, B y C. El nivel A es el más básico, el nivel B incorpora algunas características al nivel A. A su vez, el nivel C completa al nivel B con otras propiedades. Se muestran en la figura los tres niveles de implementación:

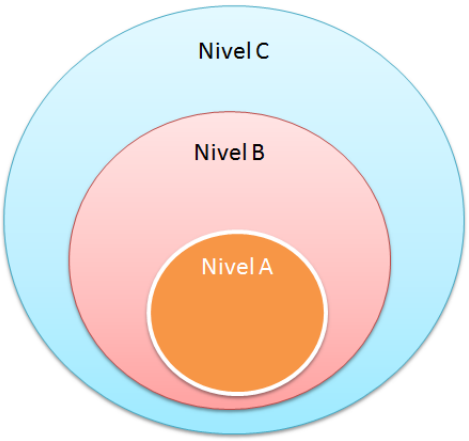


Figura 2-9. Representación lógica de los niveles A, B y C de IMS-LD

Nivel A

El nivel A es el más básico e incluye los elementos que son los pilares básicos de IMS-LD, tales como la definición de plays, actos, roles, actividades... El nivel A presenta una expresividad pedagógica limitada.

Nivel B

El nivel B completa al nivel A con dos conceptos: las **propiedades y las condiciones**. Mediante estos dos nuevos elementos es posible por ejemplo personalizar la secuencia del curso de *e-learning* para cada alumno. Las propiedades son variables que pueden usarse para almacenar por ejemplo datos relativos a la evaluación del alumno en el curso y de esa manera poder monitorizar en tiempo real el progreso del alumno [40].

Nivel C

El nivel C añade el concepto de **notificación** al nivel B. Las notificaciones son eventos que se disparan cuando sucede una determinada condición. Un buen ejemplo de una notificación es el envío automático de las notas al email de los alumnos cuando el profesor las haya publicado.

De manera gráfica se puede observar la integración de un nivel dentro de otro en la siguiente imagen:

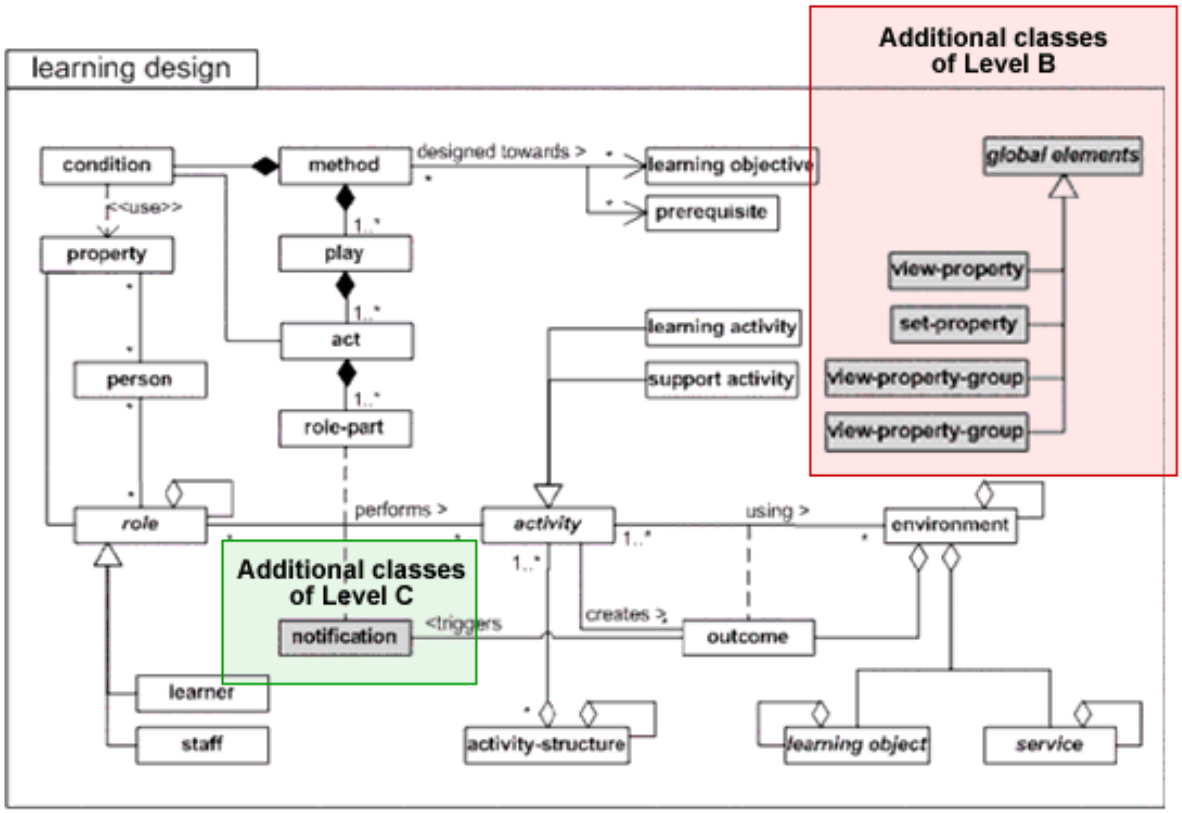


Figura 2-10. Modelo conceptual UML donde se separan los niveles A, B y C (extraído de [40])

Independientemente de la pedagogía que se esté utilizando, el proceso de aprendizaje y enseñanza se fundamenta en que unas personas asumen el rol de alumnos y otras personas asumen en rol de docentes. La finalidad del curso será la de incorporar al alumno nuevos conocimientos en un entorno concreto que está limitado en el tiempo. El entorno para el aprendizaje lo componen los *learning objects* y los servicios. Los *learning objects* son aquellos recursos que el alumno utilizará para llevar a cabo las actividades que se le propongan durante el curso. Las actividades también pueden hacer referencia a otro tipo de recursos llamados servicios, que pueden ser necesarios para realizar la actividad. Algunos ejemplos de servicios son los foros, chats, motores de búsqueda... El entorno o *environment* consiste en el conjunto de *learning objects* y *services* que serán usados durante la ejecución de las actividades.

El *method* contiene normalmente un único *play* y las condiciones (en el caso de ser nivel B o nivel C). El *play* contiene varios *acts* y cada *act* contiene varios *role-parts*. Los *role-parts* son los papeles que interpretan los actores dentro del acto en el que están definidos. Dicho de otra manera: el *role-part* asocia un rol con una actividad dentro de un acto. Una actividad, a su vez, describe aquello que se va a realizar.

Cuando se trabaja en el nivel B, las propiedades y las condiciones pueden determinar qué actividad se asocia a cada rol, de manera que se ejecutaría una planificación del curso en función del valor de las propiedades, siendo posible la adaptación de los cursos a las necesidades de los alumnos.

2.2.3. Ejecuciones, actos y actividades

El proceso de enseñanza-aprendizaje es modelado como si fuera una obra de teatro. De esta manera, una obra de teatro está formada por un conjunto de actos que se concatenan de manera secuencial. Dentro de cada acto, los actores desempeñan su papel en base a un guión establecido. Esto traducido a IMS-LD quiere decir que un *play* se compone de una serie de acts que se suceden secuencialmente, cada act tiene uno o más *role-parts* (*role-part* es el papel que interpreta cada personaje).

El siguiente esquema muestra un sencillo boceto de cómo los conceptos de *play*, *act*, *role-part* se anidan en IMS-LD:

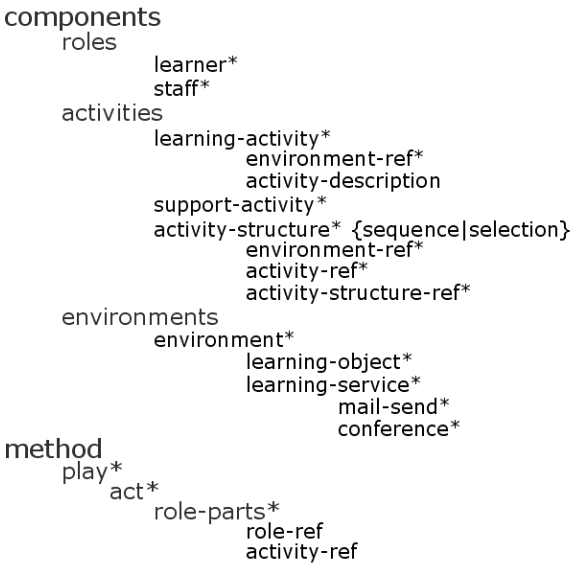


Figura 2-11. Componentes de la especificación IMS-LD

Dentro del method se especifica un *play* (o varios, en cuyo caso se ejecutarían en paralelo). Cada *play* especifica un proceso de enseñanza y aprendizaje. Un *play*, a su vez, se compone de una secuencia de actos (al menos uno), en cada acto se asignan diferentes actividades para cada rol (*role-parts*) y estas se ejecutan en paralelo. Cuando un acto finaliza, comienza el siguiente acto hasta que no queden más. Es importante destacar que, dentro de un *play*, los actos se ejecutan secuencialmente.

Los actos pueden ser usados como puntos de sincronización en los que los actores de la obra de teatro esperan a que todos acaben o por cualquier otra condición. Dicho de otro modo, los actos asocian uno o más *role-parts*, es importante tener en cuenta que los *role-parts* dentro de un acto están siempre ejecutándose en paralelo.

Cuando se activa un acto dentro de un *play*, todos los '*role-parts*' se activan. Los actores asociados a cada rol reciben una actividad o una estructura de actividades que deben realizar. La secuencia de las actividades que cada rol debe ejecutar dentro de un acto puede ser seleccionada por el alumno o bien programada por el profesor.

Las actividades se pueden asociar en estructuras formando *activity-structures*. Las actividades dentro de una *activity-structure* se pueden mostrar a los alumnos de manera secuencial o en modo selección. En el modo selección es el alumno el que elige las actividades que quiere realizar del conjunto que se le presenta, mientras que en el modo secuencial, la lista de actividades tiene que ser seguida tal y como la

ha diseñado el profesor. Esta característica permite que el alumno haga su propio camino curricular y que decida aquellas actividades que cree que son más sencillas para él o que piensa que le ayudaran a mejorar su progreso en el curso.

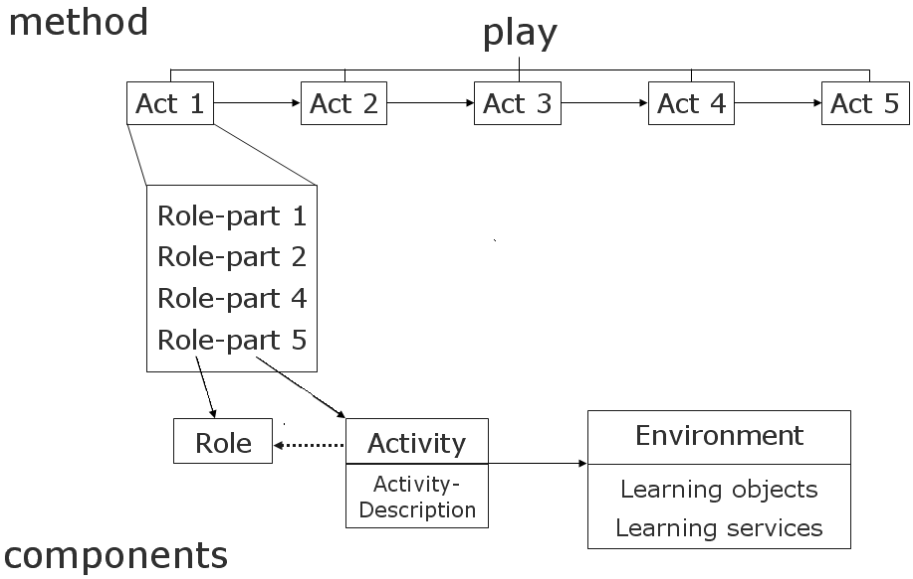


Figura 2-12. Esquema de funcionamiento de IMS-LD

Una de las características más atractivas de la especificación IMS-LD es que ha sido diseñada para soportar el mayor número posible de escenarios pedagógicos por lo que con la combinación adecuada de los diferentes elementos que la componen *(play, method, role-part*…) se puede conseguir implementar multitud de metodologías de aprendizaje [7], si bien es cierto, como se mostrará en el apartado 2.3.2, que aquellos escenarios pedagógicos que requieran la creación de bucles presentarán algunos problemas a la hora de ser implementados bajo la especificación IMS-LD.

2.3. TRABAJOS RELACIONADOS CON IMS-LD Y LAS PEDAGOGÍAS

La especificación IMS-LD fue desarrollada para dar respuesta a una necesidad hasta entonces no cubierta en las especificaciones de e-learning. La diferencia principal que incorpora IMS-LD es que supone un modelo conceptual con el cual es posible expresar múltiples aproximaciones docentes, por lo que el contenido se puede adaptar a las necesidades particulares de cada alumno y la evaluación puede estar integrada.

Dicho de otro modo, IMS-LD fue diseñado para ser capaz de implementar multitud de pedagogías y metodologías. Existen estudios [42] que demuestran la capacidad de la especificación IMS-LD para adaptarse a multitud de programaciones curriculares basadas en distintas pedagogías. Esta característica se denomina "expresividad pedagógica" y se define como la capacidad de describir la mayor cantidad de situaciones de enseñanza-aprendizaje posibles. En el siguiente apartado se profundiza en la relación que puede establecerse entre IMS-LD y las diferentes pedagogías.

2.3.1. ¿Qué relación se puede establecer entre IMS-LD y pedagogías?

Para mapear una pedagogía dentro de la especificación IMS-LD, es necesario analizarla y derivar un modelo meta-pedagógico es decir, una abstracción de la pedagogía. El motivo por el cual resulta interesante trabajar con meta-modelos es que este tipo de estructura es neutral con respecto a las diferentes aproximaciones del aprendizaje y la instrucción.

La siguiente figura, extractada de [42] muestra la relación entre el modelo pedagógico de partida y el resto de elementos involucrados hasta la creación de unidades de aprendizaje (UoL)

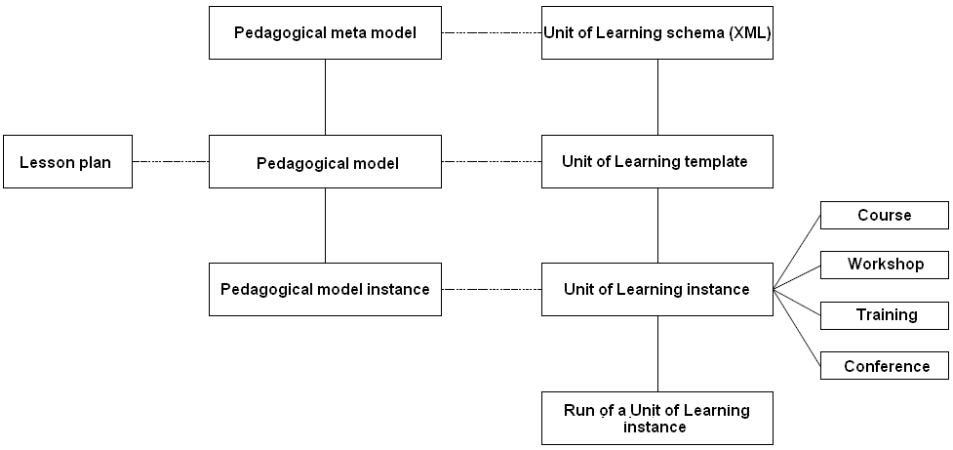


Figura 2-13. Relación entre modelos pedagógicos y Learning Design (ref. [42])

Un modelo pedagógico puede ser expresado como un meta-modelo y a su vez como una plantilla en código XML. Cada plantilla XML tendrá una estructura única, basada en las características intrínsecas de cada tipo de aprendizaje.

Cuando un modelo pedagógico se lleva a la aplicación práctica dentro del marco de unos objetivos de aprendizaje y un entorno concreto se obtiene una instancia del modelo. Paralelamente, se ejecuta una instancia de la UoL asociada.

Volviendo a la figura, aparecen a la izquierda las planificaciones del curso (*lesson plan*). Estas planificaciones son menos restrictivas y menos teóricas que los modelos pedagógicos y describen la secuencia de actividades que se sucede durante un curso.

A partir de la guía de *Best Practices and Implementation Guide* [45] de IMS-LD se puede extraer el siguiente esquema acerca de cómo generar código XML a partir de una doctrina pedagógica.

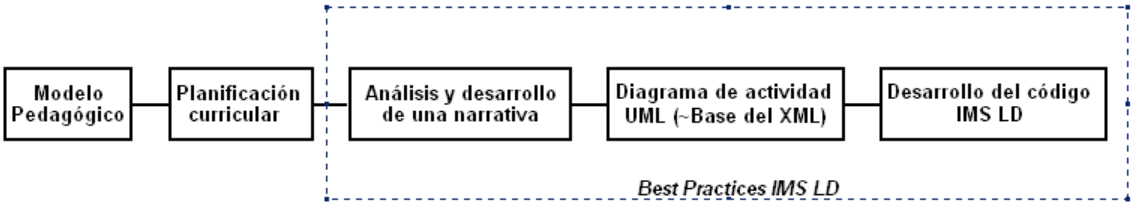


Figura 2-14. Esquema para la obtención de código IMS-LD a partir de un modelo pedagógico

Como se puede observar en la figura, a partir de un modelo pedagógico se obtiene una planificación curricular (en inglés: *lesson* plan). Esta planificación curricular describe la secuencia de conocimientos y actividades que van a ser impartidos. Además, la planificación curricular resulta ser menos restrictiva y menos teórica que el modelo pedagógico que la origina, por lo que resulta más accesible y reutilizable

[46] al utilizar un lenguaje más coloquial y menos formal. A raíz de la planificación del curso, la especificación IMS-LD recomienda la ejecución secuencial de las tareas de desarrollo de una narrativa, generar el diagrama de actividad UML y posteriormente revisar y codificar [47].

De este modo, la mayoría de los procesos educativos pueden ser descritos con suficiente solvencia mediante la especificación IMS-LD, si bien sucede que en algunos casos hay que realizar pequeños ajustes. Los servicios que proporciona LD, tales como mail, foros, conferencias... son de gran utilidad, aunque algunas situaciones académicas específicas pueden requerir algunos servicios adicionales que actualmente no están implementados.

Una de las últimas implementaciones para conseguir realizar un mapeo de las diferentes estrategias de aprendizaje ha sido realizado por el laboratorio de Riichiro Mizogouchi [48] mediante la aplicación de técnicas basadas en el concepto de ontologías para *e-learning*.

Estos estudios, mediante el enfoque de la ingeniería ontológica, proponen la creación de una base conceptual que anima a los diseñadores de cursos para poder seleccionar e integrar las estrategias adecuadas de las teorías en un contexto de instrucción. Aplicado al aprendizaje en grupo [49], esta aproximación resulta muy interesante ya que precisamente, una de las principales dificultades en el diseño de actividades de aprendizaje colaborativo es la formación de grupos adecuada.

El éxito del aprendizaje colaborativo depende directamente de la calidad en el proceso de formación de grupos. Si los grupos son correctamente asignados, los alumnos aceptan con mayor grado las actividades y el aprovechamiento del curso es por lo tanto mayor.

Sin embargo, existen múltiples factores que complican el proceso de la creación de grupos, algunos de estos factores son los conocimientos previos de los alumnos, las habilidades de aprendizaje, los roles que desempeñan y por supuesto las estrategias intrínsecas de los estudiantes a la hora de interactuar con el grupo.

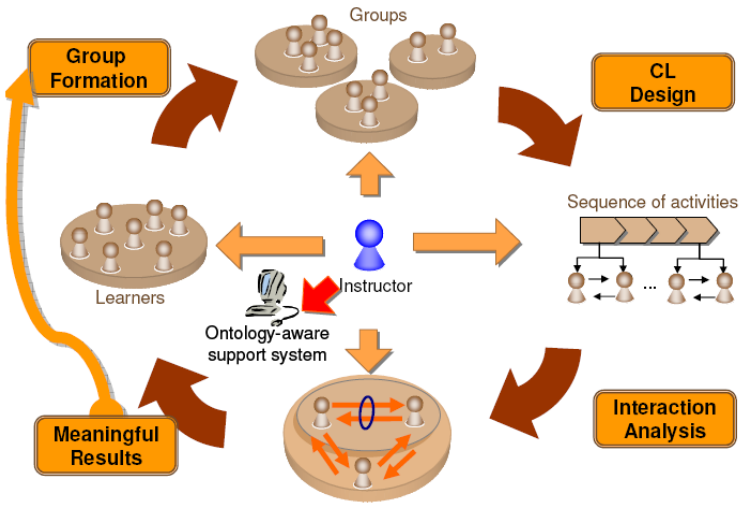


Figura 2-15. Vista completa del sistema de creación de grupos (extractado de Mizoguchi Lab. [49])

Los trabajos del grupo de Mizogouchi en este sentido, presentan una aproximación basada en ontologías que proporciona la formalización necesaria para representar el aprendizaje colaborativo y sus procesos, mientras que las teorías del aprendizaje prestan apoyo en la toma de decisiones pedagógicas como la recolección de los alumnos en grupos y la planificación del curso.

2.3.2. Limitaciones de IMS-LD

El objetivo principal de la especificación IMS-LD es el de describir escenarios pedagógicos a partir de la definición de diferentes roles y actos para los participantes.

Sin embargo, IMS-LD tiene ciertas limitaciones para la expresividad de ciertas pedagogías. Por ejemplo, no proporciona un soporte adecuado para la implementación de bucles [50], lo que supone una restricción para expresar aquellas pedagogías o metodologías basadas en el paradigma del *ensayo-error*. El hecho de que IMS-LD considere la metáfora de la obra de teatro como proceso de formación, conlleva algunas limitaciones relacionadas con el hecho de que para que un alumno acceda a un nuevo bloque formativo debe finalizar el inmediatamente anterior, al igual que para comenzar un nuevo acto en el teatro tiene que finalizar el anterior.

Se ha llegado a la conclusión [50] de que para conseguir implementar bucles en IMS-LD es necesario añadir una redundancia y complejidad excesiva al código IMS-LD además de añadir variables a introducir por parte del usuario, por ejemplo en un *textbox* durante el tiempo de ejecución. Pero incluso así, no es posible definir un bucle potencialmente infinito.

La especificación IMS-LD determina que una vez que una actividad se ha terminado, permanece completada durante el resto del *run* [7]. Este aspecto resulta ser demasiado restrictivo a la hora de representar actividades que una vez finalizadas podrían ser requeridas nuevamente a modo de refuerzo pedagógico.

Por lo tanto, es importante remarcar que el objetivo que persigue IMS-LD para poder expresar cualquier pedagogía es sólo parcialmente conseguido [50].

En el apartado 3.2 se plantea en profundidad este aspecto limitante de IMS-LD cuando se trata de codificar pedagogías cognitivistas.

Otro aspecto importante a la hora de conseguir expresividad pedagógica es la posibilidad de mostrar y ocultar diferentes actividades en función de ciertos valores o parámetros en tiempo de ejecución. Existe la siguiente lista de precedencia [7] a la hora de mostrar u ocultar actividades en IMS-LD:

- **notify** (LD Level C)
- **acts** (LD Level A)
- **sequence** (LD Level A) en una *activity-structure* en modo *sequence* se impone el valor de la *sequence* al valor de *isvisible*.
- **condition** (LD Level B) puede resetear la propiedad de *isvisible*.
- **isvisible** (LD Level A).

Por lo tanto, tendremos que tener en cuenta estas restricciones a la hora de realizar los mapeos de pedagogías a IMS-LD.

2.3.3. La actualidad de la relación entre IMS-LD y pedagogías

Pese a que la especificación de IMS-LD surge como el desafío para desarrollar una plataforma capaz de soportar la diversidad pedagógica, lo cierto es que no existe una documentación extensa acerca de cómo relacionar las distintas pedagogías clásicas con la especificación de LD.

La capacidad expresiva de IMS-LD para implementar diferentes pedagogías ha sido puesta a prueba por algunos autores. El resultado es satisfactorio ya que la especificación IMS-LD es lo suficientemente flexible como para adaptarse a la mayoría de escenarios de enseñanza-aprendizaje, si bien, es cierto que la demostración de que IMS-LD pueda soportar *cualquier* pedagogía no es definitiva.

Actualmente, una de las aplicaciones más fructífera entre pedagogías e IMS-LD ha sido la implementación de diferentes pedagogías colaborativas.. Uno de los proyectos más interesantes en torno a IMS-LD y a las pedagogías colaborativas es Collage (*http://gsic.tel.uva.es/collage*) realizado en la Universidad de Valladolid. Collage es una herramienta de autor que sirve para generar escenarios de aprendizaje colaborativo, basándose en patrones previos [51].

Otra aplicación que se ha llevado a la práctica de IMS-LD es el modelado de distintos juegos educativos. Daniel Burgos [53] es uno de los autores que más profundamente han analizado esta característica de IMS-LD llegando a interesantes conclusiones. Los juegos educativos pueden ser modelados con IMS-LD atendiendo a los niveles A y B de la especificación, es decir, con relación de actividades, roles, entornos, estructuras, método, instancias, actos, recursos y la comunicación entre ellos con propiedades y condiciones añadidas, los trabajos del grupo de Mizoguchi [49] relacionados con la aplicación de ontologías para diseño de aprendizaje son una referencia interesante en este aspecto.

Otros autores [54] han demostrado la utilidad y las ventajas de aplicar las nociones de teoría de aprendizaje constructivista a la enseñanza de disciplinas informáticas con ayuda de plataformas de e-learning. Para hacerlo, se han basado en los principios de la escuela constructivista, tales como la transformación de conocimiento por parte del alumno o la potenciación del interés del alumno a través de su autonomía y de sus intereses y han obtenido conclusiones teóricas interesantes aunque sin llegar a una realización práctica.

En resumen, se puede decir que la mayoría de los cursos bajo la especificación de IMS-LD implementados hasta el momento, han sido desarrollados por grupos de trabajo expertos en IMS-LD que han utilizado la especificación para solucionar necesidades educativas concretas.

Por otro lado, existen otros grupos de trabajo cuya actividad se ha centrado en desarrollar herramientas de autor que sean capaces de generar ficheros acorde con la especificación IMS-LD y cuya finalidad es que acaben siendo utilizadas por diseñadores de UoL. Una de las herramientas más conocidas es RELOAD [56], se trata de una potente herramienta que abarca toda la especificación pero que sin embargo está basada en términos IMS-LD y es complicada de usar por los profesores ya que para manejar Reload es necesario tener un conocimiento amplio de la especificación.

Otros acercamientos a este tipo de herramientas han sido realizados por UNFOLD, Universities of Duisburg y el departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid.

2.3.4. Actuales editores de diseño de aprendizaje

IMS-LD permite implementar aprendizaje activo, colaborativo, adaptativo, personalización de la enseñanza y otras muchas posibilidades. Según la publicación de Oberhuemer (2008) [8], las herramientas de diseño de cursos pueden clasificarse en función de las siguientes características:

- Proximidad a la especificación. Según esta clasificación, las herramientas cercanas a la especificación proporcionan herramientas basadas en términos de IMS-LD. Cuanto más cercana a la especificación es la herramienta, mayor es la cobertura de la misma pero más complicado su manejo para usuarios que desconocen IMS-LD.
- Objetivo específico o general. Esta clasificación de las herramientas se refiere al grado de flexibilidad de la herramienta es a la hora de crear escenarios pedagógicos. Cuanto más flexible es la herramienta a la hora crear cursos más general diremos que es su objetivo.

Las herramientas más conocidas para generar unidades de aprendizaje IMS-LD son CopperCore [57], CopperAuthor [58], Reload [56], Sled [59] o algunas otras extraídas del proyecto europeo UNFOLD Project [63] y del Learning Network for Learning Design [65]. Como se citaba en el anterior apartado, Collage [51] es una particularización de Reload para modelar escenarios de aprendizaje colaborativo. Una de las últimas herramientas desarrolladas (2010) para la creación de unidades de aprendizaje es Prolix Graphical Learning Modeller [64], un editor de uso intuitivo para desconocedores de IMS-LD.

En el apartado de conclusiones (pag. 97) se muestra un diagrama del posicionamiento de estas herramientas en función de los criterios de Oberhuemer. A continuación se recoge la información más relevante sobre algunas de estas herramientas de autor para el diseño de unidades de aprendizaje.

En la siguiente figura se puede ver una toma de pantalla del editor de Reload [56], uno de los más completos y cercanos a la especificación. Como puede verse, se trata de una herramienta codificada como aplicación de escritorio en la que el usuario va introduciendo los diferentes parámetros IMS-LD para el curso que se está creando.

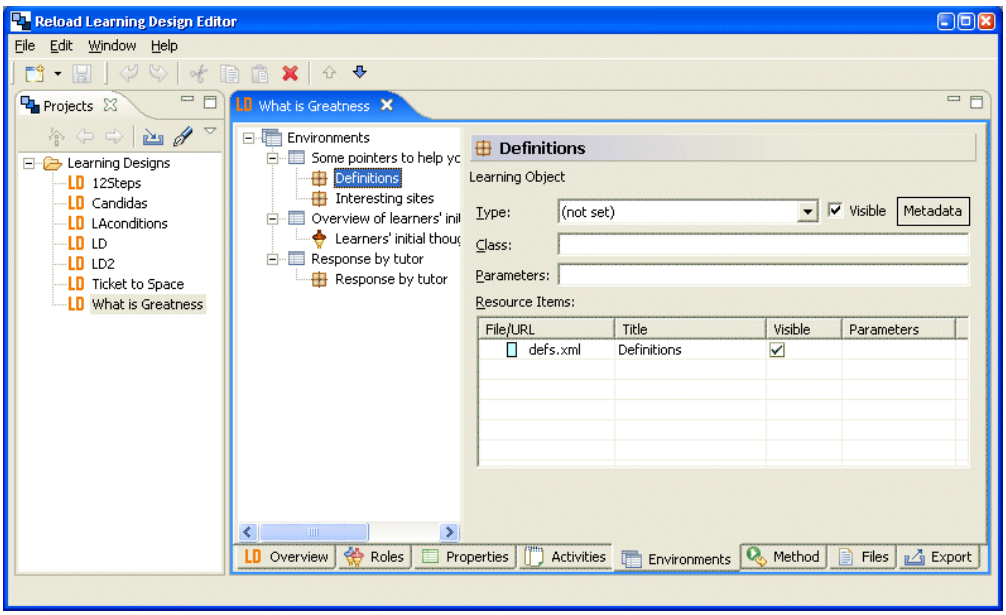


Figura 2-16. Vista del Learning Design Editor de Reload

El inconveniente más importante de estos editores es que requieren que los autores tengan un buen conocimiento de la especificación. Existen otros editores de alto nivel que no imponen esta exigencia. Por ejemplo el sistema MOT+ [66] incorpora funciones que permiten exportar en formato Diseño de Aprendizaje, si bien es cierto que MOT+ hace uso de la nomenclatura y los términos de IMS-LD y resulta algo complicado de manejar para personas ajenas a la especificación.

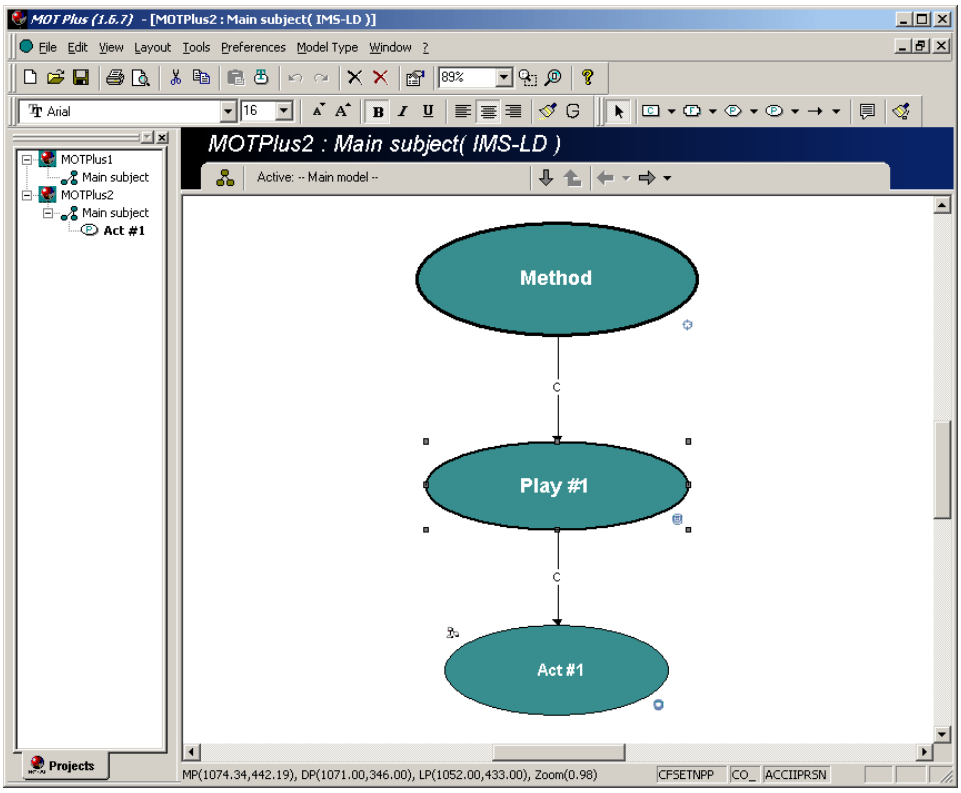


Figura 2-17. Vista del editor de cursos MOT+ [66]

Prolix GLM [64] proporciona una interfaz intuitiva para modelado de cursos, lo que reduce en gran medida la complejidad de la especificación IMS-LD y permite crear UoLs a partir de conexiones de elementos gráficos en terminología cercana a los diseñadores de cursos.

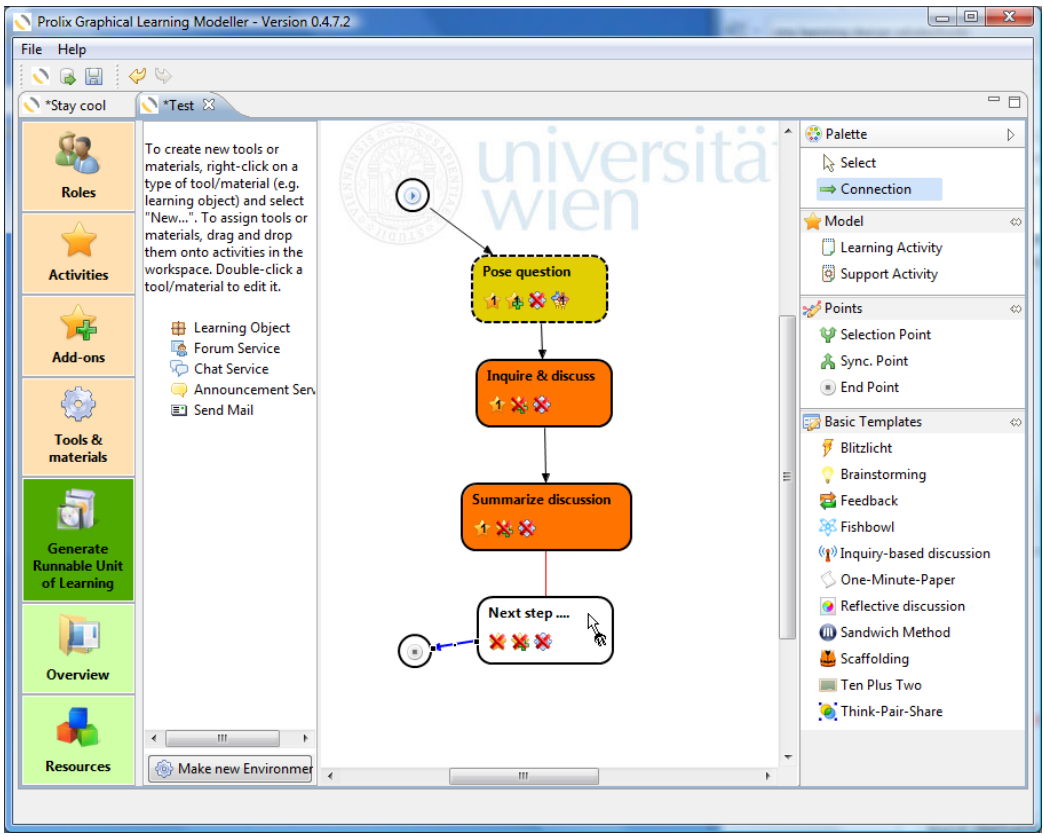


Figura 2-18. Vista del software para edición de cursos de Prolix GLM [64]

Otro conocido editor de cursos es LAMS [67] que dispone de una interfaz, arrastrar y soltar, fácil de usar que permite combinar diferentes actividades de aprendizaje dentro de la estructura de una lección. Se hace cada vez más necesario el desarrollo de herramientas orientada a la pedagogía y lo más manejables posible para que los profesores puedan utilizarlas con sencillez y conseguir con ello planificaciones de curso más versátiles.

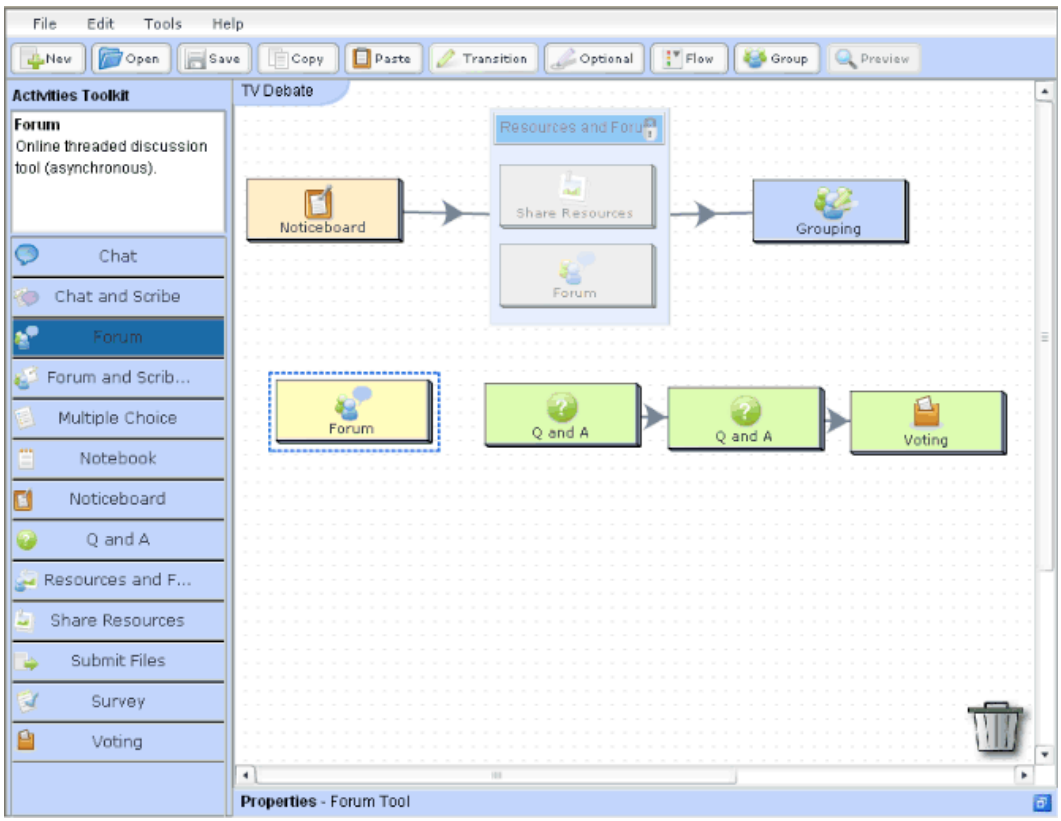


Figura 2-19. Vista del software para edición de cursos de LAMS

Durante el año 2005 se sucedieron encuentros y actividades dentro del marco internacional con el objetivo de dar a conocer y potenciar las posibilidades de la especificación IMS-LD. Uno de los asuntos más importantes sobre los que se trabaja en la actualidad es cómo obtener una UoL en IMS-LD bien estructurada a partir de una planificación curricular centrada en educación presencial. El otro asunto fundamental es cómo crear estas UoL de una manera sencilla incluso para personas cuyo perfil no sea técnico.

Una de las últimas herramientas surgidas para dar respuesta a la necesidad de facilitar la creación de cursos bajo el estándar IMS-LD ha sido desarrollada por el TenCompetence [68] y se trata de una herramienta de autor basada en el IDE Eclipse [69] que propone una interfaz moderna y realmente completa para generar cursos basados en IMS-LD:

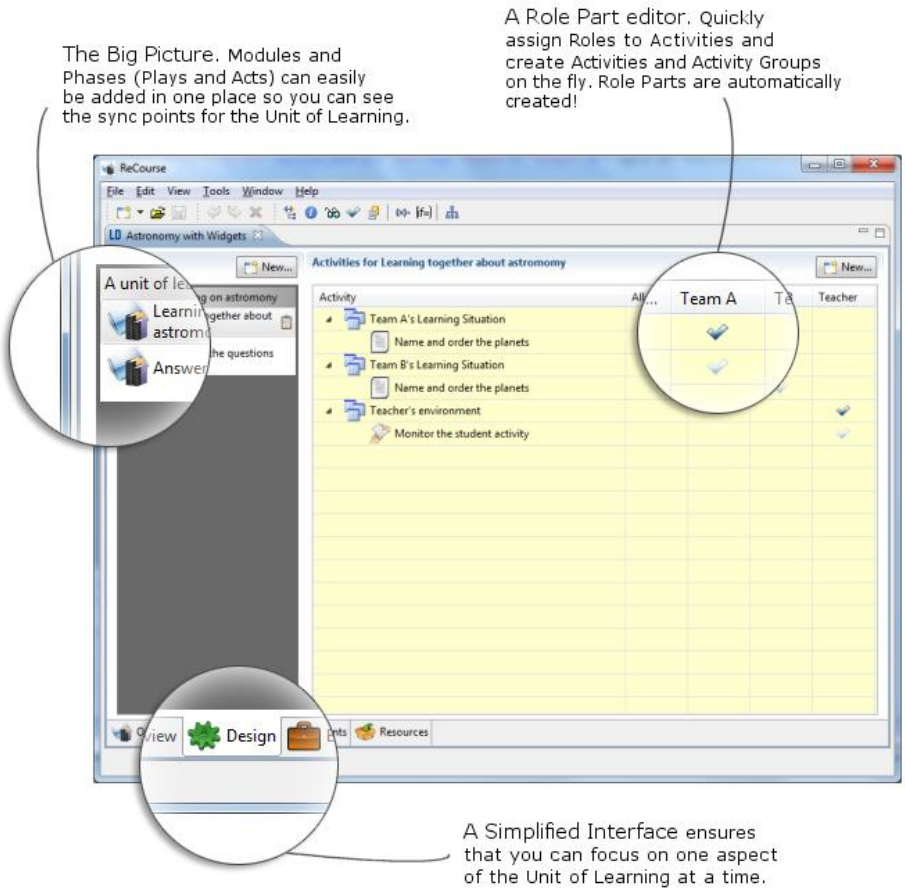


Figura 2-20. Vista del editor de learning design ReCourse de TenCompetence

La herramienta de autor de TenCompetence se llama ReCourse y proporciona una interfaz amigable para la creación de cursos con IMS-LD aunque exige al autor el conocimiento de la especificación IMS-LD y resulta algo complicada de manejar.

En cualquier caso, se trata de una potente y flexible herramienta para incluso poder conectar con un servidor de CopperCore para crear nuevas *run* a partir de los diseños de aprendizaje modelados con la herramienta.

La herramienta de autor que se ha implementado en este proyecto fin de carrera mejora a las anteriores en el aspecto de que el usuario realmente no tiene que introducir ningún parámetro de IMS-LD para generar los cursos, simplemente manejando conceptos pedagógicas, el usuario de PedaLea puede llegar a generar cursos completos bajo la especificación IMS-LD completamente funcionales.

2.3.5. Los reproductores de e-learning (RTEs)

Desde que en febrero de 2003 se aprobara IMS-LD se han ido desarrollando un cierto número de herramientas relacionadas con la especificación, tanto editores como reproductores. La complejidad de la especificación IMS-LD conlleva que la aplicación que la esté ejecutando sea bastante compleja.

Uno de los reproductores de e-learning más conocidos es Coppercore [57]. Se trata de un reproductor *opensource* capaz de ejecutar un IMS-LD, lo suficientemente complejo como para permitir el seguimiento en la evolución de los alumnos. En la siguiente imagen se puede ver una toma de pantalla durante la ejecución de un curso en Coppercore:

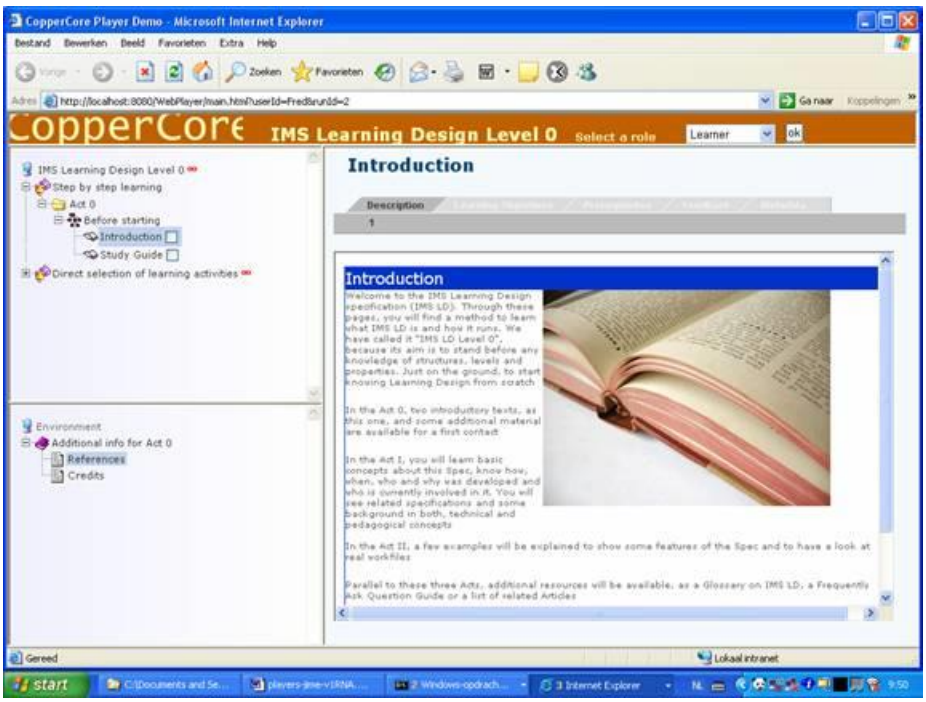


Figura 2-21. Vista del Player CopperCore

Coppercore tan sólo dispone de una interfaz sencilla pero el hecho de que sea de código abierto facilita que otros desarrolladores puedan añadir interfaces más complejas para dar lugar a reproductores más completos. Por ejemplo, a mediados del año 2005 Reload [56] presentó su Learning Design Player 2.0.0 que incorpora el motor de CopperCore. Actualmente esta herramienta se encuentra en la versión 2.1.3.

Reload consiste en una interfaz mejorada para interactuar con el motor de CopperCore. En la siguiente figura puede verse cómo el navegador para visualizar el curso está embebido dentro de la ventana de la aplicación Reload.

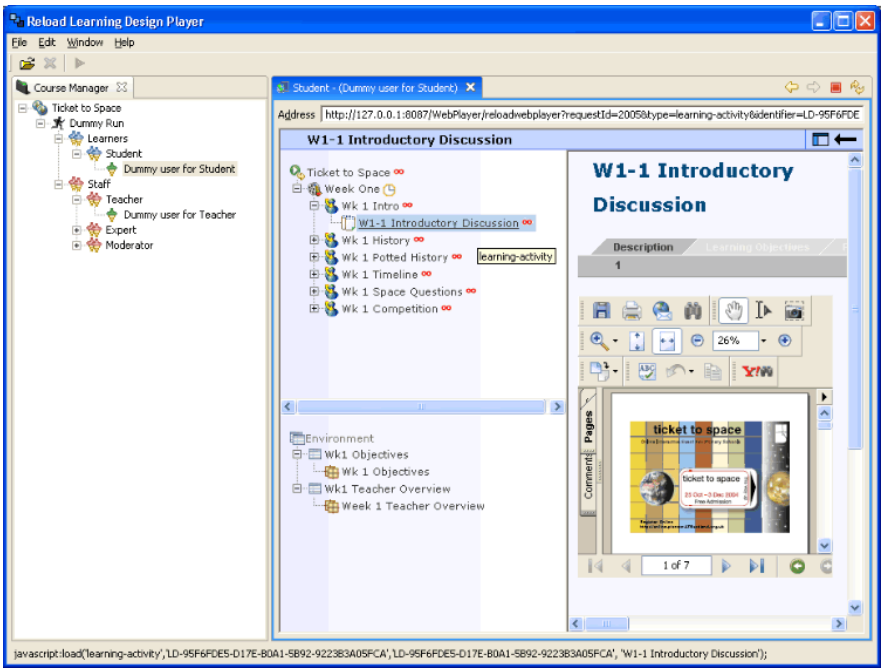


Figura 2-22. Vista del Player Reload basado en el motor de CopperCore

Las características principales de Reload Learning Design Player son las siguientes:

- Envuelve el motor de CopperCore en una interfaz sencilla y fácil de gestionar.
- Despliega automáticamente Coppercore en un servidor JBoss AS [70].
- Lee un *Learning Design* y automáticamente genera una ejecución por defecto del curso con usuarios ficticios sin la necesidad de interactuar con la línea de comandos como sí exige CopperCore.

Otro entorno de ejecución es GRAIL [71] (*Gradient-lab RTE for Adaptive IMS-LD in .LRN*), un entorno de ejecución de IMS-LD implementado como parte de .LRN. Ha sido realizado por el Laboratorio Gradient de la Universidad Carlos III de Madrid (UC3M) y soporta IMS-LD en los niveles A, B y C.

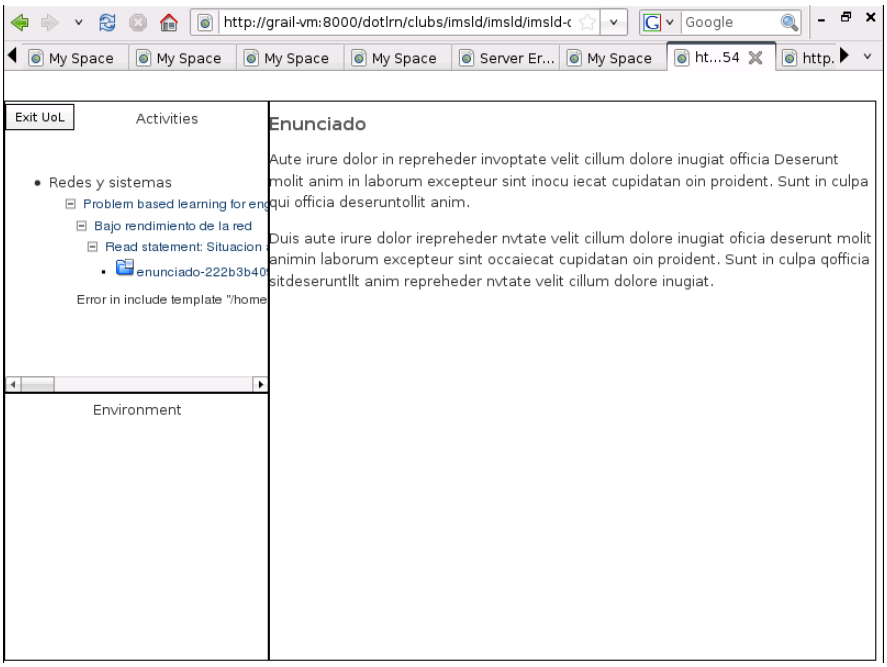


Figura 2-23. Vista del Player de IMS LD GRAIL

La ventaja de este *player* de IMS-LD es que está empotrado dentro de un LMS, lo que le permite usar servicios extra como foros, chats, calendario, etc.

Para simplificar el trabajo de instalación del sistema, es posible acceder a la descarga de una máquina virtual [72] configurada y operativa para poder acceder de una manera rápida y sencilla a la utilización de la plataforma GRAIL.

2.4. CUESTIONES ABIERTAS Y CONTRIBUCIÓN DE ESTE PROYECTO

La implementación de la herramienta de autor que motiva este proyecto es un acercamiento de la especificación IMS-LD a los docentes y creadores de cursos.

De entre las herramientas de autor analizadas en el apartado 2.3.4, las más conocidas (Reload o ReCourse, por ejemplo) tienen como denominador común que están directamente acopladas a conceptos IMS-LD. Este hecho dificulta que un docente desconocedor de la especificación IMS-LD pueda generar un curso con cierta raíz pedagógica con alguna de estas herramientas.

La herramienta de autor de este proyecto fin de carrera, PedaLea, está basada en términos pedagógicos que son sencillos de comprender para cualquier docente. Durante la creación de un nuevo curso, el usuario no se encuentra con ningún término basado en IMS-LD, por lo que la herramienta de autor implementada en este proyecto consigue uno de sus objetivos primitivos ya que aísla por completo la complejidad de la especificación a los creadores de cursos.

El aspecto indiscutiblemente positivo de la ausencia de términos IMS-LD durante la creación del curso es fundamentalmente la gran cantidad de usuarios que serán capaces de crear cursos con una cierta orientación pedagógica sin tener nociones de IMS-LD en absoluto. El aspecto controvertido surge del mismo punto ya que las pedagogías implementadas en la herramienta están desarrolladas de manera que no se puede modificar su naturaleza, por lo que ¿qué hará un docente que quiera añadir alguna particularidad a las pedagogías desarrolladas por la herramienta? Se plantean para responder a la última cuestión dos opciones y ambas pasan por la necesidad de que el docente cuente o bien con conocimientos técnicos o con el soporte de un equipo técnico.

La primera solución al problema de la personalización de las pedagogías en esta herramienta es retocar los ficheros XML de IMS-LD para añadir las particularidades que quiera introducir el docente en el curso concreto. Esto supone sin duda una complicación a la hora de generar el curso y la posibilidad de cometer errores pero por otro lado, la herramienta de autor le habrá aportado una base sólida y válida en términos de IMS-LD sobre la que poder introducir alguna ligera variación.

La segunda solución y probablemente la más interesante es la posibilidad de generar su propia pedagogía a medida. Una vez que un docente diseña cómo quiere impartir un curso, debería recurrir a un equipo técnico que basándose en el *Manual del Desarrollador* (Apéndice C) generara las clases y configuraciones necesarias para que la aplicación pudiera incorporar esa nueva metodología. Para ello, el desarrollador cuenta con varias pedagogías de ejemplo ya desarrolladas y además con un desarrollo pensado ser continuado con futuros trabajos. Se han integrado en el desarrollo las tecnologías más comúnmente usadas en aplicaciones web Java (Spring, Struts, Tiles, Hibernate…) y siguiendo los patrones de diseño que son más utilizados en la industria del desarrollo de aplicaciones web Java.

Hay al menos dos importantes cuestiones que quedan abiertas después del desarrollo de esta herramienta. Una de estas cuestiones ha sido abordada en el apartado de limitaciones de IMS-LD (apartado 2.3.2) y se trata de la limitación en la expresividad pedagógica de IMS-LD. La restricción de que una actividad que el alumno ha realizado quede marcada como realizada durante el resto del curso, impide que ciertas aproximaciones pedagógicas y metodológicas basadas en el paradigma de *ensayo y error* puedan ser descritas bajo la especificación IMS-LD.

Otra de las cuestiones y quizá aún más importante que queda manifiestamente en el aire tras las pruebas realizadas con esta herramienta de autor es la ausencia de un RET de IMS-LD completo y sólido en el que los cursos puedan ser instanciados cubriendo los niveles A, B y C de la especificación.

Para un mismo curso, se han detectado diferencias significativas durante el *runtime* entre GRAIL y Coppercore [57]. Algunas de estas diferencias están relacionadas con el hecho de mostrar u ocultar ciertas actividades en algunos momentos o con la asignación de propiedades. A lo largo de los siguientes apartados se irán comentando las discrepancias encontradas entre ambos RTEs.

3. MODELADO DE DIFERENTES PEDAGOGÍAS EN IMS-LD

Un archivo de diseño de aprendizaje descrito acorde a la especificación IMS-LD se compone básicamente de dos grandes bloques: *Components* y *Method*. Los componentes son de tipo: *Roles*, *Properties*, *Activities* y *Environments*. El *Method* sería una descripción de cómo se deben combinar los diferentes componentes que se están empleando.

Desde un punto de vista práctico, para IMS-LD todo modelado de aprendizaje está basado en un *Method* que asocia varias *Activities* a los diferentes *Roles* que participan en un curso [7]. A partir de aquí, los *Environments* se definen como un entorno de servicios y recursos de aprendizaje que estarán disponibles para la realización de las actividades del curso. Para la particularización de los aprendizajes y la implementación de pegagogías más complejas hay que recurrir al uso de *Properties, Conditions* y *Notifications*. En el apartado 2.2.2 se muestran los diferentes niveles de IMS-LD y el alcance de cada uno de ellos.

IMS-LD no proporciona soporte específico para ninguna pedagogía, de hecho se trata de una especificación amplia y capaz de expresar, como veremos a continuación, diferentes modelos pedagógicos a partir de la configuración de sus diferentes elementos.

En el siguiente apartado se mostrará la manera en la que los diferentes elementos que componen IMS-LD se han configurado para implementar las diferentes pedagogías que se han desarrollado para este proyecto. En el Apéndice B – Manual de usuario, se muestra la forma en la que el usuario debe interactuar con la aplicación web para crear UoL con la orientación pedagógica deseada.

3.1. PEDAGOGÍA LINEAL

Se trata de la implementación de una metodología que muestra a los alumnos una concatenación de actividades que se mostrarán de manera consecutiva a medida que el alumno va dando por finalizada la lectura o realización de las diferentes actividades.

Para la implementación de esta metodología tendremos en cuenta solamente el rol del estudiante. Para hacer entrega de los *learning objects* a los alumnos se empleará un elemento *activity-structure* que agrupe las diferentes *learning-activities* que van a componer el curso. Por lo tanto, los componentes a utilizar en esta pedagogía serían los mostrados en la siguiente figura:

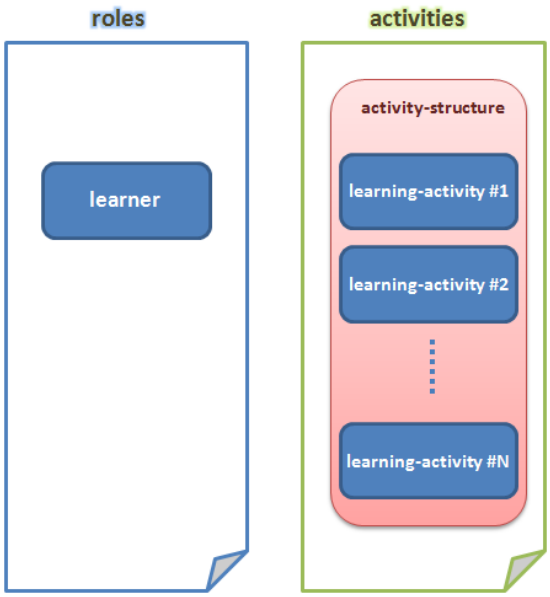


Figura 3-1. Componentes IMS-LD utilizados en la pedagogía lineal

El nivel de IMS-LD requerido para la implementación de este tipo de pedagogía es A.

La *activity-structure* se configura en modo *sequence* de modo que a medida que el alumno vaya terminando *learning-activities* irá accediendo a las siguientes hasta completar el curso.

El *method* para esta pedagogía también será muy sencillo. Consistirá en definir un único *play* que contendrá a su vez un único *act*. En este único acto solamente será necesario definir un *role-part* que asignará al rol de los alumnos la única *activity-structure* definida en el listado de componentes. Esquemáticamente se puede ver el method para la pedagogía lineal en la siguiente figura:

Figura 3-2. Esquema del *method* para la pedagogía lineal

3.2. PEDAGOGÍA COGNITIVISTA DE THORNDIKE

Se trata de una implementación pedagógica basada en el refuerzo cognitivo. De esta manera, la metodología de un curso basado en esta aproximación pedagógica consistirá en mostrar una serie de materiales a los alumnos en un bloque que acabará en una evaluación de algún tipo para reforzar los conocimientos adquiridos en el bloque de contenidos. Según el resultado de la evaluación dada por el profesor se permitirá a los alumnos continuar con el siguiente bloque o tendrán que repetir el bloque de nuevo. Esta es la clave del refuerzo cognitivo sobre la que se basa esta pedagogía.

En este punto es importante recordar que según la especificación IMS-LD las *learning-activities* sólo pueden ser realizadas una vez. La repetición de las *learning-activities* no está soportada por la especificación de forma inherente por lo que tendremos que recurrir a alguna estrategia para emular este comportamiento cíclico.

La solución propuesta para este caso es replicar las actividades de manera que cuando un alumno requiere repetir un bloque de contenidos lo que se hace realmente es mostrarle una copia de contenido que tiene que repetir. De esa forma, se podrían llegar a crear bucles finitos que para este caso hemos limitado a tres iteraciones.

Los componentes IMS-LD requeridos para la evaluación de los contenidos serán, por un lado los roles de *learner* y *staff*. Adicionalmente serán necesarios un conjunto de *properties* para la que el profesorado pueda determinar la promoción de los alumnos así como las actividades que deben mostrarse u ocultarse para realizar la anterior implementación de los bucles. Se empleará por lo tanto el nivel B de IMS-LD para la implementación de esta pedagogía.

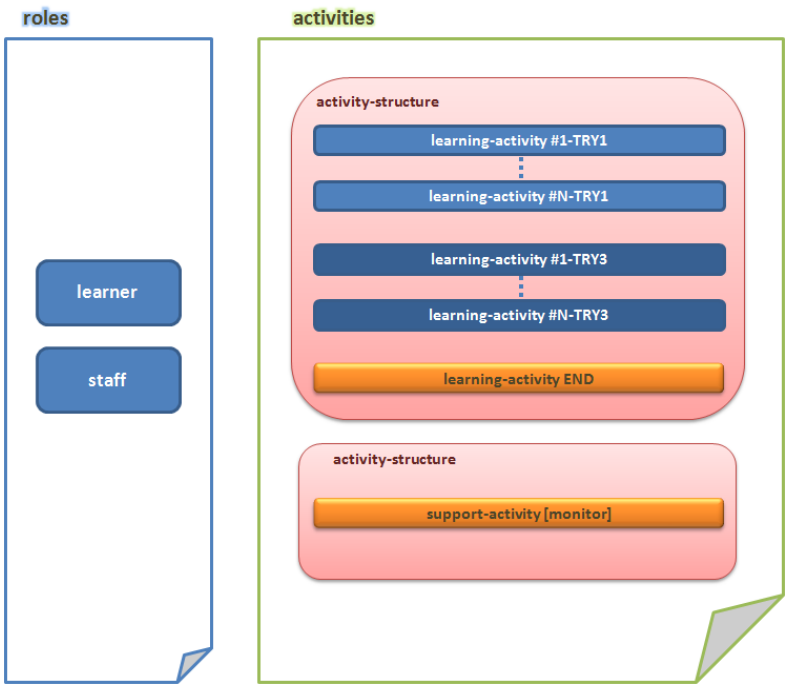


Figura 3-3. Componentes *roles* y *activities*

En la figura anterior, se puede observar la manera en la que se implementará el refuerzo cognitivo mediante la simulación de bucles de tres iteraciones. Cada actividad del curso se replicará tres veces dentro de un bloque de *activity-structure* que a su vez terminará con una actividad de tipo “END” que

crea automáticamente la aplicación y simplemente informa al usuario de que ha completado el bloque de actividades y pasará al siguiente bloque (si este existiera).

Existirá otra *activity-structure* que servirá para que el personal en el rol de *staff* pueda por un lado acceder a los contenidos que los alumnos están enviando a través de la plataforma de formación así como determinar su itinerario a lo largo del curso. Este tipo de evaluaciones se realizan mediante un servicio *monitor*. Los servicios *monitor* deben ir referenciados dentro de un *environment* que se mostrará asociado a una actividad o a un conjunto de ellas. En la siguiente figura se muestran los elementos de tipo *environment* utilizados así como los elementos de tipo *property*:

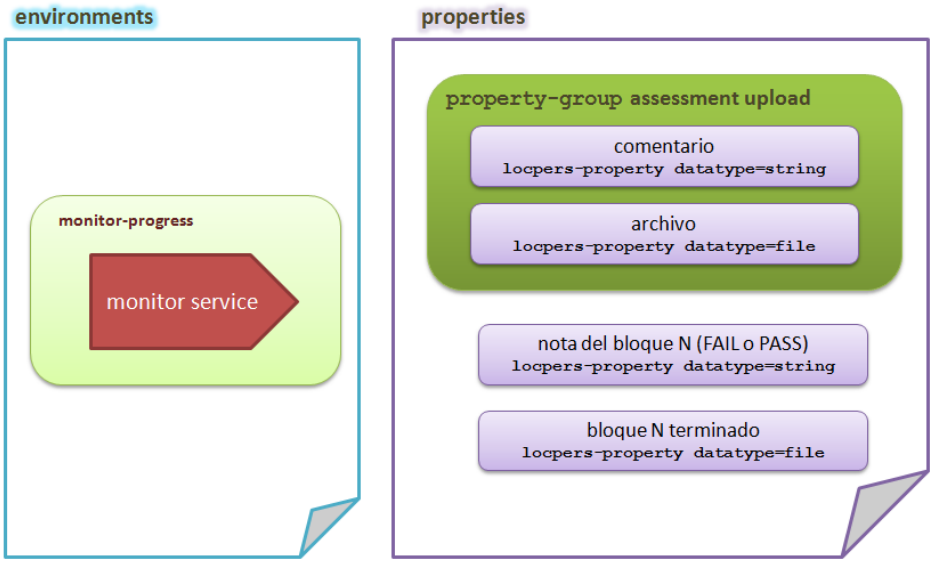


Figura 3-4. Componentes de *environments* y *properties* en la pedagogía Thorndike

Las *properties* son imprescindibles en este tipo de pedagogía por dos motivos. El primer motivo es que serán utilizadas para almacenar los contenidos que los alumnos envían a través de la plataforma. Estos contenidos (*assesment upload*) se componen de unos comentarios y un fichero que se adjuntará a través de un formulario web que se desplegará en tiempo de ejecución. El segundo motivo es que serán utilizadas para realizar la muestra o la ocultación de las *learning-activities* a lo largo del curso.

Para que los usuarios puedan enviar comentarios y ficheros, la aplicación crea automáticamente unos ficheros "upload-X.xml" que se referencian desde las *learning-activities* desde las que se espera que el usuario realice algún tipo de entrega.

```
    <resource identifier="R-3-2" type="imsldcontent" href="upload-3-2.xml">
      <file href="upload-3-2.xml" />
    </resource>
```

El contenido de estos ficheros XML de upload es el siguiente:

```
<?xml version="1.0"?>
<!-- Microondas, file automatically generated. [http://pld-project.net] -->
<html xmlns:ld="http://www.imsglobal.org/xsd/imsld_v1p0"
xmlns="http://www.w3.org/1999/xhtml">
<head>
<title/>
</head>
<body>
<p><font color="#000000" size="3">Please, upload your file and comments: </font></p>
<p><ld:set-property-group ref="Assessment-3" property-of="self" view="title-value"/></p>
<hr></hr>
<p><font color="#000000" size="3">These are your comments and your assesment</font></p>
```

```
<p><ld:view-property-group ref="Assessment-3" property-of="self" view="title-value"/></p>
</body>
</html>
```

De ese modo, cuando un alumno llega al final de un conjunto de temas y se le propone un ejercicio para que envíe unos resultados, estos resultados se guardarán en la propiedad *assesment*. El profesor a través del servicio monitor, al que podrá acceder a través del *environment*, podrá acceder a leer los comentarios y a descargar el fichero para su evaluación. Posteriormente podrá seleccionar de un elemento desplegable si la calificación para el alumno es *PASS* o *FAIL*.

En el caso de que el profesor elija la opción de *PASS*, se mostrará la primera actividad del siguiente bloque al usuario. Si por lo contrario el profesor elige la opción de *FAIL* se le mostrará la segunda réplica de la primera actividad del bloque actual.

Este mecanismo de comportamiento a lo largo del curso está descrito en el elemento *method* de IMS-LD del que se muestra un esquema en la siguiente figura:

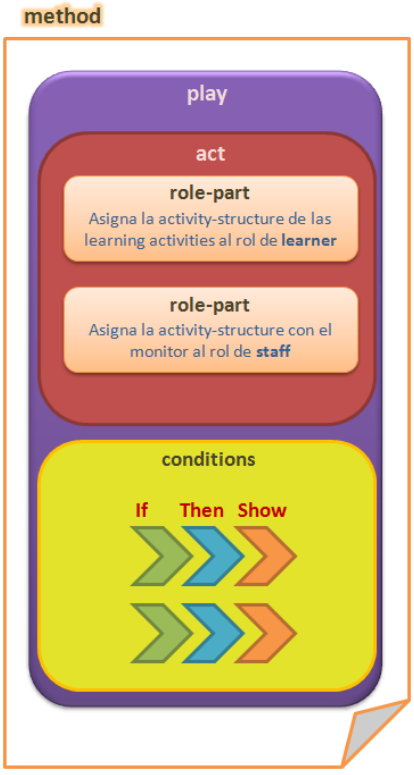


Figura 3-5. Figura del *method* para la pedagogía Thorndike

En el *method* de esta pedagogía existirá un único elemento *play* que contendrá a su vez un único elemento *act*. En este acto se realizarán dos asignaciones de *role-part*, una de las asignaciones se realizará para que los alumnos dispongan del conjunto de *learning-activities* que se eligieron en la fase de diseño del curso y la otra asignación se realiza para que usuarios en el rol de *staff* tengan acceso a la *support-activity* desde cuyo *environment* se podrá acceder al servicio *monitor* para monitorizar tanto los *uploads* de los alumnos como sus itinerarios formativos a lo largo del curso.

La implementación de esta pedagogía pone de manifiesto una restricción significativa que presenta la especificación IMS-LD a la hora de implementar aprendizajes por repetición que requieran la ejecución de bucles.

3.3. APRENDIZAJE BASADO EN LOS 4 PASOS DE POLYA

Esta metodología está basada en la repetición de los mismos cuatro pasos para resolver cualquier tipo de problema: comprensión del problema, planteamiento de un plan, ejecución y revisión de resultados. En el apartado de ejecución del plan, se permitirá al creador del curso elegir si desea o no recibir por parte de los alumnos el ejercicio resuelto para su valoración. Al igual que en la pedagogía anterior, estos *uploads* realizados por parte de los alumnos se guardarán en propiedades locales personales de IMS-LD, por lo que para la implementación de esta pedagogía emplearemos el nivel B.

Durante el tiempo de ejecución de este tipo de curso, los roles involucrados en esta serán por un lado los alumnos que serán los protagonistas del proceso formativo y por otro lado los profesores que estarán encargados de visualizar (si los hubiese) los resultados de los problemas que hayan enviado los alumnos. En la siguiente figura se puede ver la composición de *roles* y *activities* considerada para la implementación de esta pedagogía:

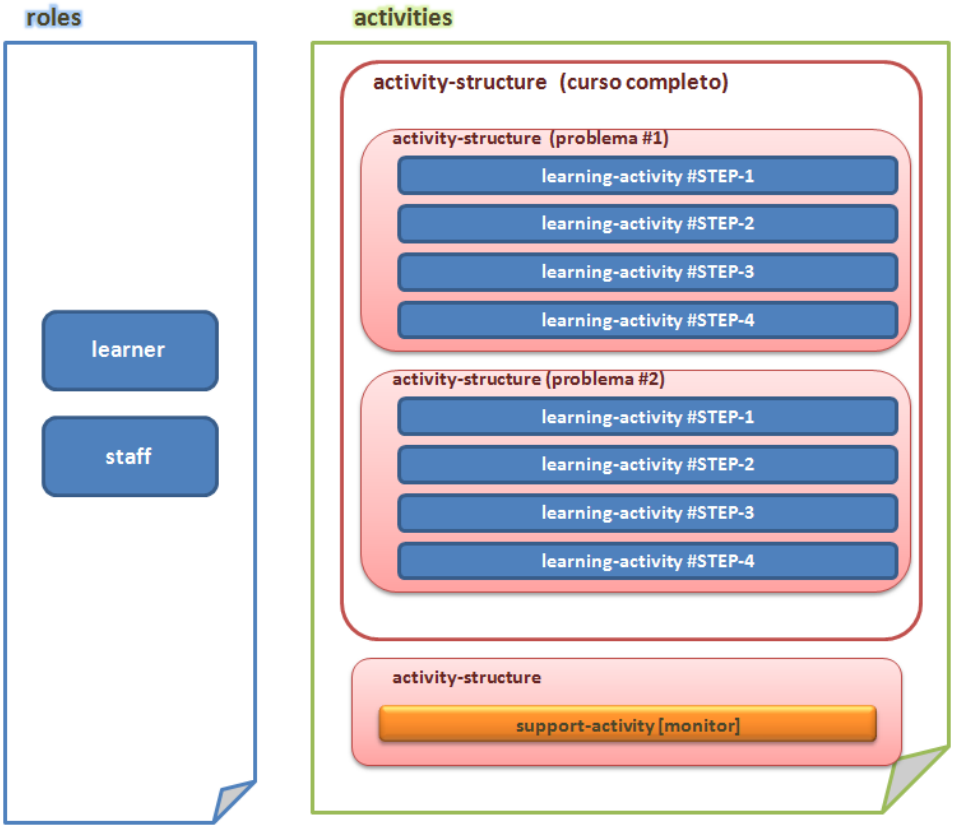


Figura 3-6. Componentes *roles* y *activities* para la pedagogía de Polya

Como se observa en la figura anterior, cada problema se compone de cuatro *learning-activities* que se agrupan dentro de una *activity-structure.* La condición de *complete-activity* por cada actividad está

definida a *user-choice* para que cada alumno pueda dedicar a cada una de las cuatro fases del problema el tiempo que le convenga.

Se determina que cada *activity-structure* que contiene un problema va a tener una estructura de tipo *sequence* para que el alumno vaya visualizando en orden cada uno de los pasos que debería completar para resolver el problema planteado.

En conjunto de problemas se agrupa a su vez en una *activity-structure* global (en la figura anterior: curso completo) que será asociada a los alumnos en el elemento *role-part* del *method*.

Por otro lado, existirá otra *activity-structure* que contendrá una única *support-activity* desde cuyo *environment* se accede al servicio *monitor* para poder revisar los archivos y comentarios enviados por los alumnos en el paso 3 durante el tiempo de ejecución. En la siguiente figura se muestran los componentes de IMS-LD tipo *environment* y tipo *properties* para poder implementar esta operativa:

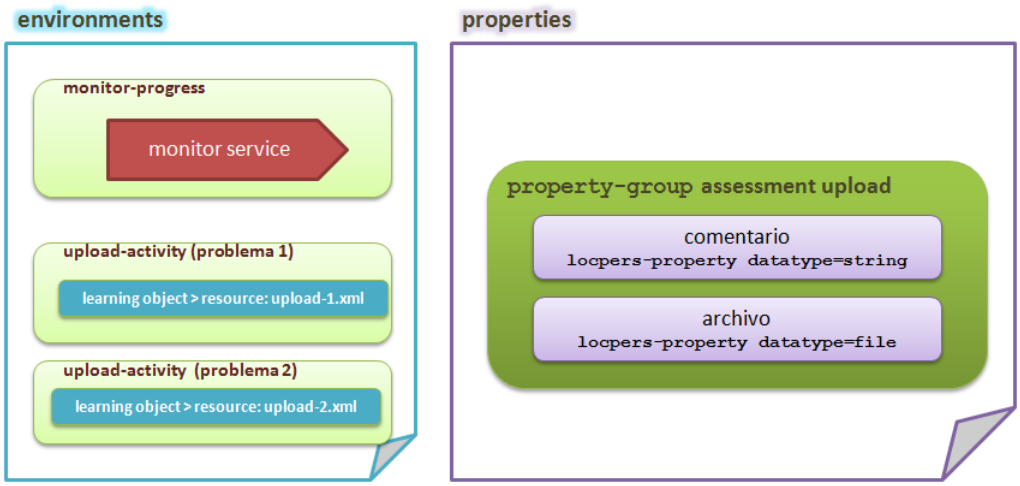


Figura 3-7. Componentes de tipo *environment* y *properties* para la pedagogía de Polya

Por un lado, los comentarios y archivos enviados por los alumnos se almacenarán como propiedades *locpers-property* que serán agrupadas en un *property-group*.

Durante el diseño del curso el profesor puede habilitar a los alumnos la opción de subir a la plataforma la solución que han desarrollado para el problema. Se trata por lo tanto de un aspecto complementario que se puede añadir al paso 3 pero que no es sustitutivo de la *learning-activity* que puede existir en ese punto. Para lograr la convicencia de una *learning-activity* con un recurso de *upload* lo que se empleará es una referencia al recurso de *upload* en el *environment* del paso 3. Por este motivo, existirán tantos *environments* de tipo *upload* como problemas haya con la opción de subir la solución a la plataforma por parte de los alumnos.

El servicio *monitor* permite visualizar el contenido de todos los *upload* realizados por usuario. No es necesario por lo tanto tener un servicio monitor por cada problema con *upload*. La aplicación genera de forma automática por cada curso un fichero *"monitor.xml"* que contiene las referencias adecuadas para poder acceder a todos los comentarios y ficheros que los usuarios hayan enviado durante el tiempo de ejecución. A continuación se muestra un ejemplo del contenido de este fichero:

```
<?xml version="1.0"?>
    <!--
            Calculo numerico, file automatically generated. [http://pld-project.net]
    -->
<html xmlns:ld="http://www.imsglobal.org/xsd/imsld_v1p0"
xmlns="http://www.w3.org/1999/xhtml">
    <head>
            <title />
    </head>
    <body>
            <p>
                    <font color="#000000" size="3">Comments and assessment for
                            problem 1: Teorema del valor medio</font>
            </p>
            <p>
                    <ld:view-property-group ref="Assessment-6"
                            property-of="supported-person" view="value" />
            </p>
            <hr></hr>
            <p>
                    <font color="#000000" size="3">Comments and assessment for
                            problem 2: Teorema de la integral</font>
            </p>
            <p>
                    <ld:view-property-group ref="Assessment-7"
                            property-of="supported-person" view="value" />
            </p>
            <hr></hr>
    </body>
</html>
```

Para completar el mapeo a IMS-LD de esta pedagogía solamente falta por conocer el elemento *method*. Para esta pedagogía se ha implementado un *method* que contiene un *play* que a su vez contiene un *act* que contiene dos *role-part*, uno para los *learners* y otro para el *staff*. Se muestra gráficamente en la siguiente figura:

Figura 3-8. Vista esquemática del *method* para la pedagogía de Polya

3.4. APRENDIZAJE BASADO EN LOS 3 PASOS DE MASON, BURTON Y STACEY

Esta metodología de aprendizaje consiste en separar la resolución de un problema en tres pasos diferentes. Durante el tiempo de diseño del curso, el profesor podrá crear cursos con el número de problemas que considere necesario y deberá añadir para cada uno de estos problemas un recurso (archivo o URL) por cada uno de los 3 pasos (*Entry, Attack* y *Looking back*).

Se trata de una simplificación del caso de los cuatro pasos de Polya y se emplea para su mapeo en IMS-LD una implementación similar a la del punto 3.3.

En la siguiente figura se muestran los *roles* y *activities* empleados para el mapeo de esta pedagogía.

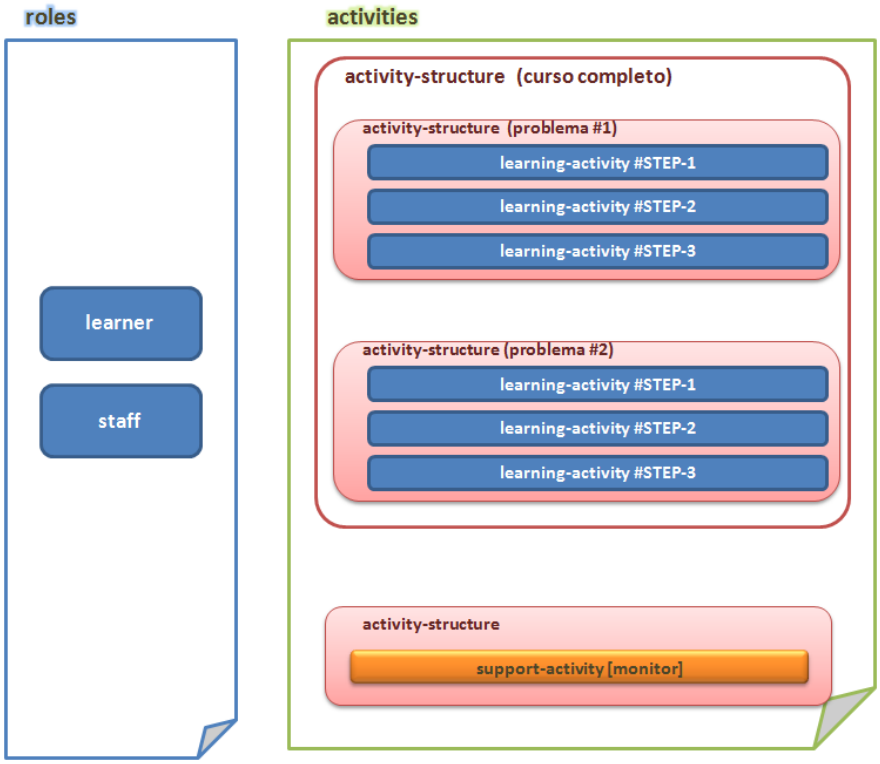


Figura 3-9. Vista de *roles* y *activities* para la pedagogía de Mason

Como se muestra en la figura anterior, se crean dos grandes *activities-structures,* una de ellas alberga tantas *activity-structure* como problemas se hubieran definido y la otra contiene una *support-activity* para la monitorización por parte del *staff* de los resultados de los *learner.*

Como también se puede apreciar en la figura, el número de *learning-activities* dentro de cada problema corresponde con los tres pasos de esta pedagogía.

El resto de los componentes y del *method* tienen un comportamiento común con la pedagogía de Polya del punto 3.3.

3.5. APRENDIZAJE BASADO EN PROBLEMAS PARA INGENIERÍA

La *Escola de Engenharia de São Carlos* [24] ha desarrollado una metodología para resolver problemas en siete fases para favorecer el aprendizaje de los alumnos. En el punto 2.1.5.4 de este documento (página 14) se profundiza sobre este tipo de aprendizaje y cada uno de los pasos que lo componen. Como se

muestra más adelante se emplea el nivel B de la especificación IMS-LD para implementar esta pedagogía.

Durante el tiempo de diseño del curso, el profesor tan sólo tiene que elegir un enunciado para el problema y los recursos adicionales que requerirá el alumno para resolverlo, será la aplicación web la encargada de generar los siete pasos para la resolución del problema.

Pese a que, en líneas generales, esta pedagogía consiste en la resolución de problemas en una serie de pasos, la implementación es muy diferente a las mostradas en los puntos 3.3 y 3.4. En este caso al profesor solamente se le pide como *input* en enunciado de un problema amplio y opcionalmente recursos adicionales que los alumnos puedan necesitar. A partir de esa información se genera un curso completo que contiene foros de debate, recursos de monitorización de alumnos y otras propiedades características que se explican a continuación.

Para implementar esta pedagogía debe existir un rol de *staff* y otro rol de *learner*. Tanto el profesor como alumnos podrán participar en los foros de debate sobre el planteamiento y la solución del problema así como en la creación de un replanteamiento del enunciado del problema. En la siguiente figura se muestran los componentes de IMS-LD de tipo rol y de tipo *activity* que se han empleado.

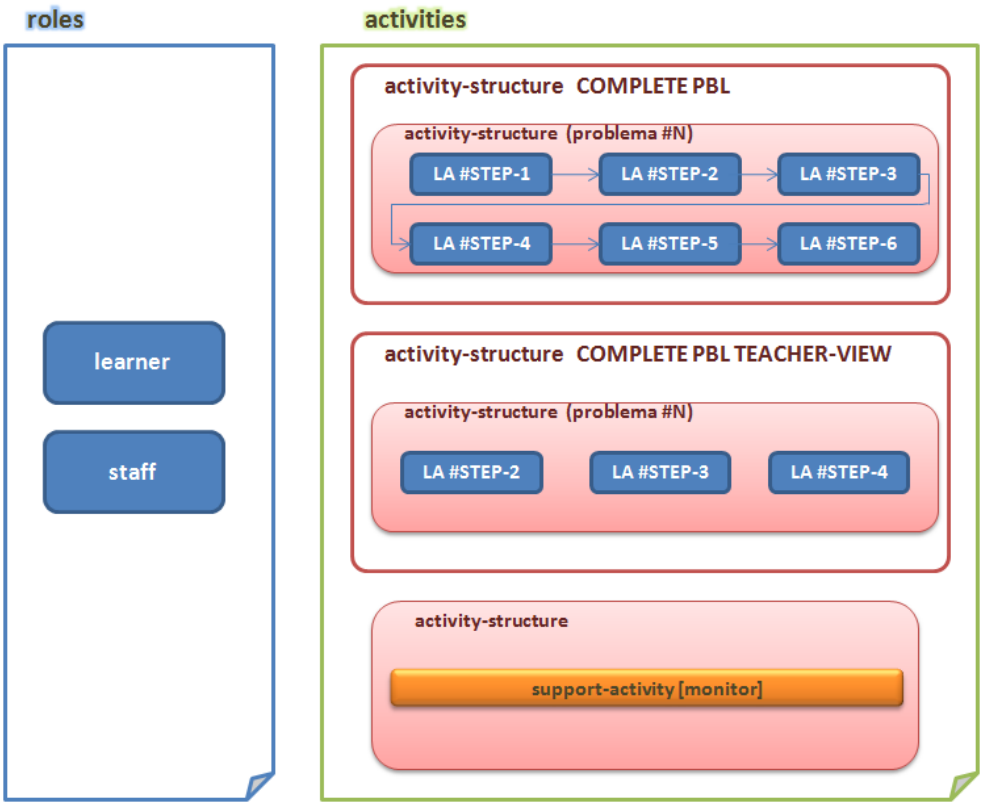


Figura 3-10. Componentes de tipo *roles* y *activities* para Problem Based Learning

En la figura anterior se puede ver que las *learning-activities* del curso se agrupan en *activity-structures*. De este modo, cada *learning-activity* es un paso en la resolución del problema y existirán tantos *activity-structures* de agrupación de pasos como problemas contenga el curso. Todos los problemas a su vez se agrupan en otra *activity-structure* (en la figura anterior *COMPLETE PBL)*.

En esta pedagogía el profesor debe poder monitorizar a los usuarios en dos planos diferentes. Por un lado e igual que sucedía en anteriores pedagogías, el docente monitorizará los *assesment* entregados por los alumnos y consistentes en unos comentarios y un fichero adjunto. Adicionalmente a esta monitorización, el profesor participará en tres de los seis pasos de la pedagogía. Concretamente, el profesor estará presente y podrá participar y moderar los foros de debate acerca de *"What do we know?"* y *"What do we need to know?",* pasos 2 y 4 respectivamente.

El paso 3 es especialmente interesante para el profesor. En este punto de la pedagogía los alumnos desarrollarán de manera común una reformulación del enunciado del problema. El profesor podrá visualizar este enunciado y cuando lo considere oportuno podrá darle un valor final consiguiendo un doble objetivo: por un lado ofrecer a los alumnos una reformulación orientada y por otro lado, permitir que a los alumnos se les muestre el paso 4 de le pedagogía.

En la figura anterior, a la agrupación de las *learning-activities* que se mostrarán al profesor se le ha llamado *COMPLETE PBL TEACHER-VIEW*.

La posibilidad de que cada alumno pueda enviar un enunciado del problema en el paso 3 y un *assessment* en el paso 6 sugiere la necesidad de utilizar propiedades de tipo *locpers-property.* Además la reformulación del enunciado en el paso 3 sugiere la existencia de una propiedad común a todos los usuarios tanto *staff* como *learner*. Se empleará para ello una propiedad *loc-property*. En la siguiente figura se muestra el resto de componentes IMS-LD para esta metodología.

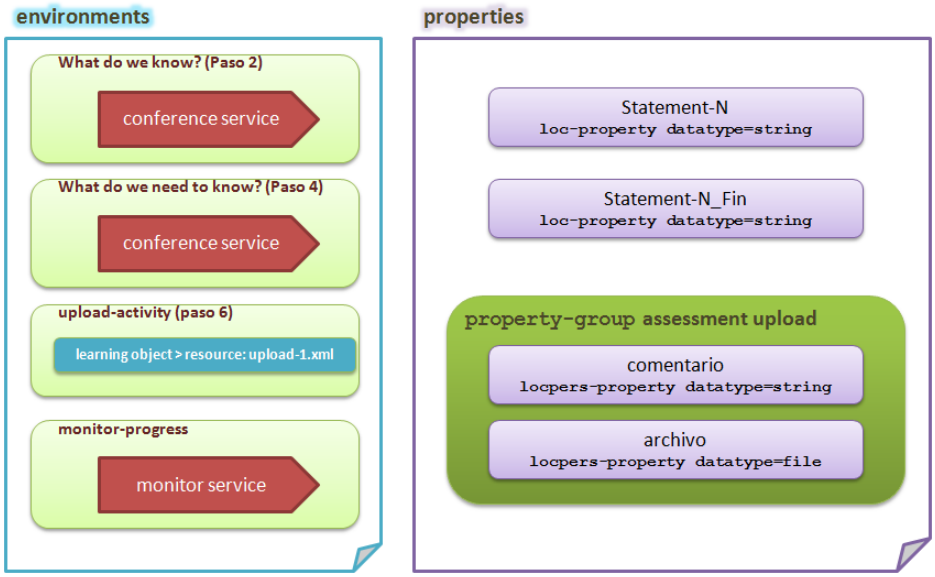


Figura 3-11. Componentes de tipo *environment* y *properties* para Problem Based Learning

La propiedad "*Statement-N_Fin"* será de tipo *enumeration* y será utilizada para condicionar que se muestre a los alumnos la actividad del paso 4 para el problema N.

Los *environments* son especialmente importantes en esta pedagogía. Para los alumnos, los *environments* cumplirán tres funciones principalmente. Primeramente serán los puntos desde el que se podrá acceder a los foros de debate de los pasos 2 y 4. En segundo lugar será el lugar desde donde podrá acceder al material complementario de autoestudio del paso 5. Y por último será el lugar desde

donde podrá acceder a la actividad para el envío de la solución del problema. Para los profesores, el *environment* será también utilizado para el acceso a los foros y por otro lado para el acceso a la monitorización de los estudiantes.

Para esta pedagogía, el método contendrá un único *play* que a su vez contendrá un único acto. Los *role-part* asignados en este acto se muestran en la siguiente figura:

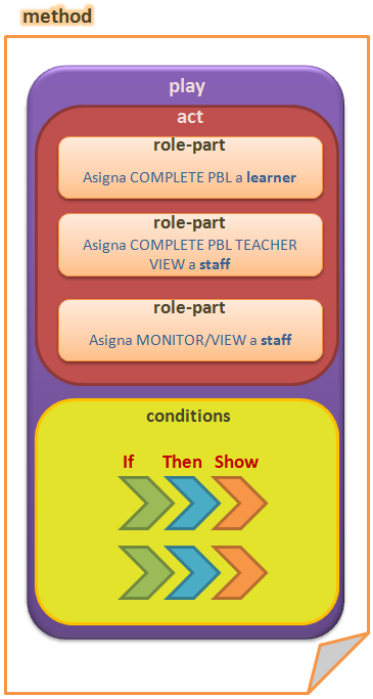


Figura 3-12. Vista esquemática del *method* para la pedagogía *Problem Based Learning*

Las condiciones para el caso de la pedagogía basada en problemas deben permitir que al alumno se le muestre la actividad 4 (y las sucesivas) cuando el profesor pone a valor *"YES"* la propiedad *loc-property* que indica que el enunciado reformulado del problema está validado y finalizado por el profesor: *Statement-N_fin.*

El siguiente bloque es código IMS-LD generado por la aplicación donde se puede ver en detalle la manera en la que las actividades 4, 5 y 6 son mostradas cuando la propiedad *Statement-1_fin* toma valor "YES".

```
<imsld:conditions>
    <imsld:if>
            <imsld:is>
                    <imsld:property-ref ref="Statement-1_fin" />
                    <imsld:property-value>YES</imsld:property-value>
            </imsld:is>
    </imsld:if>
    <imsld:then>
            <imsld:show>
                    <imsld:learning-activity-ref
                            ref="LA-s4-e38b3bf9-d32c-40dc-921c-0dd8c09e9ef9" />
            </imsld:show>
    </imsld:then>
    <imsld:if>
            <imsld:is>
                    <imsld:property-ref ref="Statement-1_fin" />
                    <imsld:property-value>YES</imsld:property-value>
            </imsld:is>
    </imsld:if>
    <imsld:then>
            <imsld:show>
```

```
                    <imsld:learning-activity-ref
                            ref="LA-s5-e38b3bf9-d32c-40dc-921c-0dd8c09e9ef9" />
            </imsld:show>
    </imsld:then>
    <imsld:if>
            <imsld:is>
                    <imsld:property-ref ref="Statement-1_fin" />
                    <imsld:property-value>YES</imsld:property-value>
            </imsld:is>
    </imsld:if>
    <imsld:then>
            <imsld:show>
                    <imsld:learning-activity-ref
                            ref="LA-s6-e38b3bf9-d32c-40dc-921c-0dd8c09e9ef9" />
            </imsld:show>
    </imsld:then>
    <imsld:if>
            <imsld:is>
                    <imsld:property-ref ref="Statement-1_fin" />
                    <imsld:property-value>YES</imsld:property-value>
            </imsld:is>
    </imsld:if>
    <imsld:then>
            <imsld:hide>
                    <imsld:class class="set_statement_problem1" />
            </imsld:hide>
    </imsld:then>
</imsld:conditions>
```

Al estar las *learning activities* 4, 5 y 6 dentro de una *activity-sctructure* de tipo *sequence*, el hecho de poner sobre ellas la condición de *show =true* no las muestra directamente sino que las marca como mostrables, de manera que se mostrarán cuando llegue su turno dentro de la secuencia de actividades de la estructura. Para que esto funcione de esta manera, será necesario que las *learning-activities* presenten en su definición *isvisible=false.* En el siguiente bloque de código se muestra la definición de una de estas *learning-activities.*

```
<imsld:learning-activity identifier="LA-s5-e38b3bf9-d32c-40dc-921c-0dd8c09e9ef9"
      isvisible="false">
      <imsld:title>Self study. P1</imsld:title>
      <imsld:environment-ref
              ref="E-Study-S5-e38b3bf9-d32c-40dc-921c-0dd8c09e9ef9" />
      <imsld:activity-description>
              <imsld:item identifier="I-s5-e38b3bf9-d32c-40dc-921c-0dd8c09e9ef9"
                      identifierref="RES-STEP5-P1" />
      </imsld:activity-description>
      <imsld:complete-activity>
              <imsld:user-choice />
      </imsld:complete-activity>
</imsld:learning-activity>
```

3.6. APRENDIZAJE BASADO EN PROYECTOS CON ROLES PARALELOS

Se trata de una metodología que consiste en dividir a los alumnos en grupos para que puedan realizar tareas en paralelo. De este modo, durante la fase de diseño del curso se elegirán cuales son las partes del proyecto a ejecutar en paralelo y dentro de cada parte cuáles son los roles existentes.

Para cada uno de los roles, el diseñador del curso elegirá una serie de recursos que se mostrarán a los alumnos de ese rol durante el tiempo de ejecución.

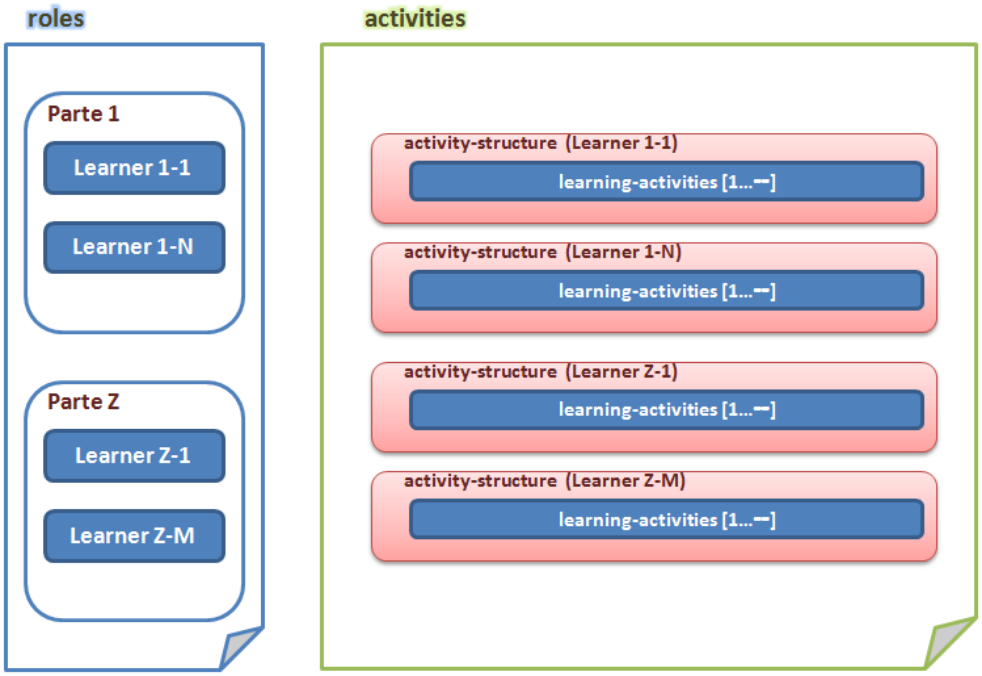


Figura 3-13. Componentes de tipo *roles* y *activities* para aprendizaje basado en proyectos con roles en paralelo

Como se puede ver en la figura anterior, el proyecto constaría de Z partes y dentro de cada parte habría un número de roles (N y M respectivamente). Para cada rol se declara una *activity-structure* que contendrá las *learning-activities* asignadas a cada uno de los roles.

Para esta pedagogía se llegan a utilizar los componentes IMS-LD de *environments* o *properties*.

El *method* para esta pedagogía consiste en un único *play* con un único acto en el cual hay tantos *role-parts* asignados como roles se han dado de alta en las diferentes partes del proyecto. Se muestra en la siguiente figura.

Figura 3-14. *Method* para aprendizaje basado en proyectos con roles en paralelo

3.7. APRENDIZAJE BASADO EN PROYECTOS EN CICLO DE VIDA

Esta metodología se basa en la posibilidad de que los alumnos participen en las diferentes fases de un proyecto. Por ejemplo, para el caso de un desarrollo de software se pasaría por las fases de "Toma de requisitos", "Análisis", "Implementación", "Pruebas"…

Durante el tiempo de diseño del curso el profesor decide cuáles serán las fases del proyecto y qué recursos están asignados a cada una de estas fases.

De forma automática, la aplicación genera por cada fase del proyecto un rol de líder de fase y un rol de integrante. Una fase del proyecto termina cuando el alumno líder completa la última actividad. Por defecto, esta última *learning-activity* será un *assesment* en el que el líder de la fase subirá a la plataforma el *output* resultante del trabajo de todos los integrantes en esa fase. El output de la fase anterior se entrega a los alumnos de la siguiente fase. Se consigue así que el trabajo desarrollado en la fase N sea el punto de partida para los alumnos de la fase N+1.

Para gestionar y controlar todo el proyecto, se crea un rol de *staff* a modo de jefe de proyecto, que tendrá visibilidad a través de un elemento *monitor* a los diferentes *output* que vayan surgiendo de cada fase del proyecto. En la siguiente figura se muestran los distintos *roles* y *activities* empleados para la implementación de este tipo de aprendizaje:

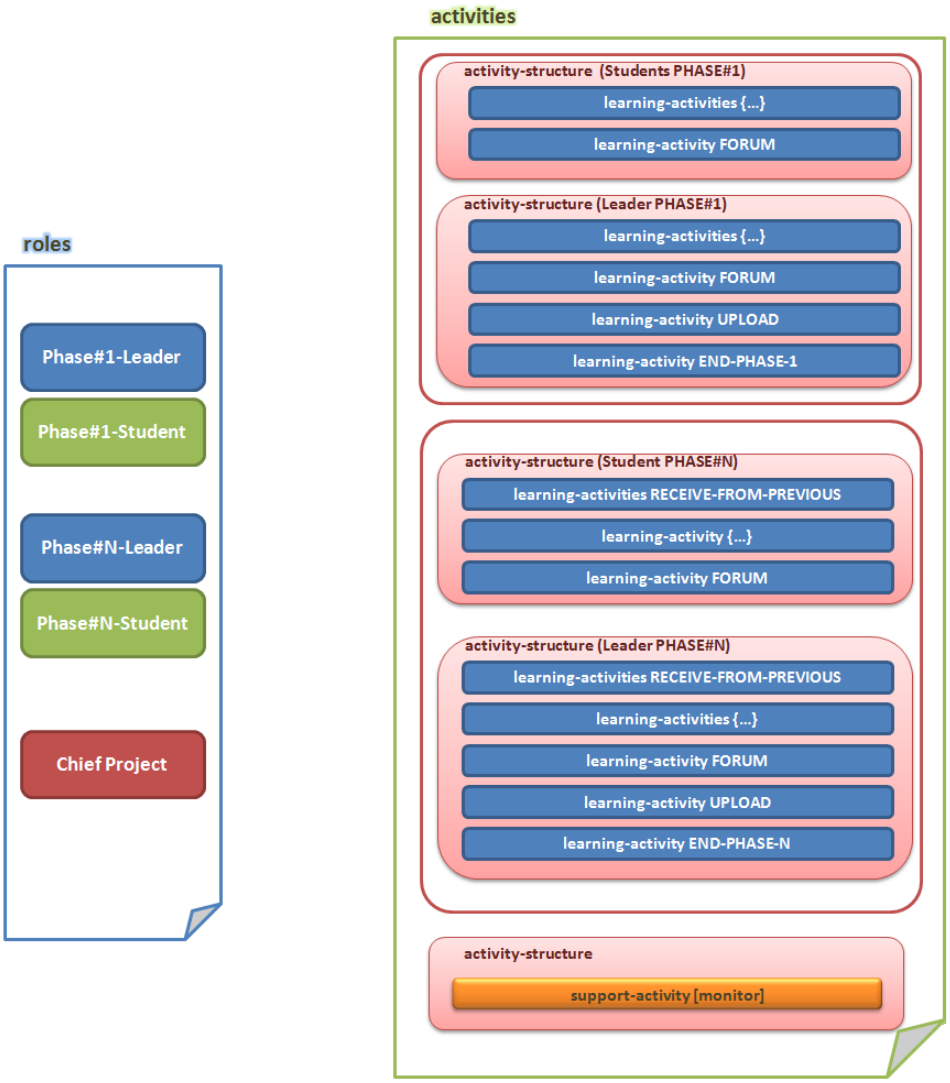


Figura 3-15. Roles y *activities* para la pedagogía orientada a proyectos en ciclo de vida

Como se aprecia en la figura anterior, para un proyecto de N fases existirán (2N+1) roles. Esto es: un rol de líder y otro de *student* por cada fase y adicionalmente un jefe de proyecto.

En cuanto a las actividades, cada fase tendrá una o varias *learning-activities* que se eligen durante el tiempo de creación del curso. En la figura es lo que se ha denominado *"learning-activities{…}"*.

Para los estudiantes y para los líderes de las fases se crean diferentes *activities-structures* que contendrán adicionalmente actividades para albergar foros de debate y para facilitar la subida y descarga de archivos de la plataforma de formación. Estos archivos serán los elementos de conexión entre los integrantes de las diferentes fases del proyecto.

En cada una de las fases se genera un foro de debate para que los integrantes y líder puedan comunicarse, compartir el desarrollo y colaborar para la realización de esa fase del proyecto. Estos foros o servicios de *conference* estarán referenciados desde *environments* disponibles para las actividades de cada fase.

Para poder realizar el paso de los *output* de una fase a la siguiente, será necesario contar con propiedades de tipo *loc-property* que almacenen tanto un comentario sobre el resultado del proyecto así como un archivo resultado de los desarrollos de la fase. Para simplificar la gestión de estas *loc-properties*, las agruparemos en *property-groups.* En el siguiente fragmento de código se ve la forma en la que la aplicación agrupa las propiedades:

```
<imsld:loc-property identifier="Comment-1">
     <imsld:title>Your comments</imsld:title>
     <imsld:datatype datatype="string" />
</imsld:loc-property>
<imsld:loc-property identifier="File-1">
     <imsld:title>Your upload</imsld:title>
     <imsld:datatype datatype="file" />
</imsld:loc-property>
<imsld:property-group identifier="Upload-1">
     <imsld:title>Properties Group</imsld:title>
     <imsld:property-ref ref="Comment-1" />
     <imsld:property-ref ref="File-1" />
</imsld:property-group>
```

Se muestra en la siguiente figura se muestran los componentes de *environment* y *properties* de IMS-LD para esta pedagogía.

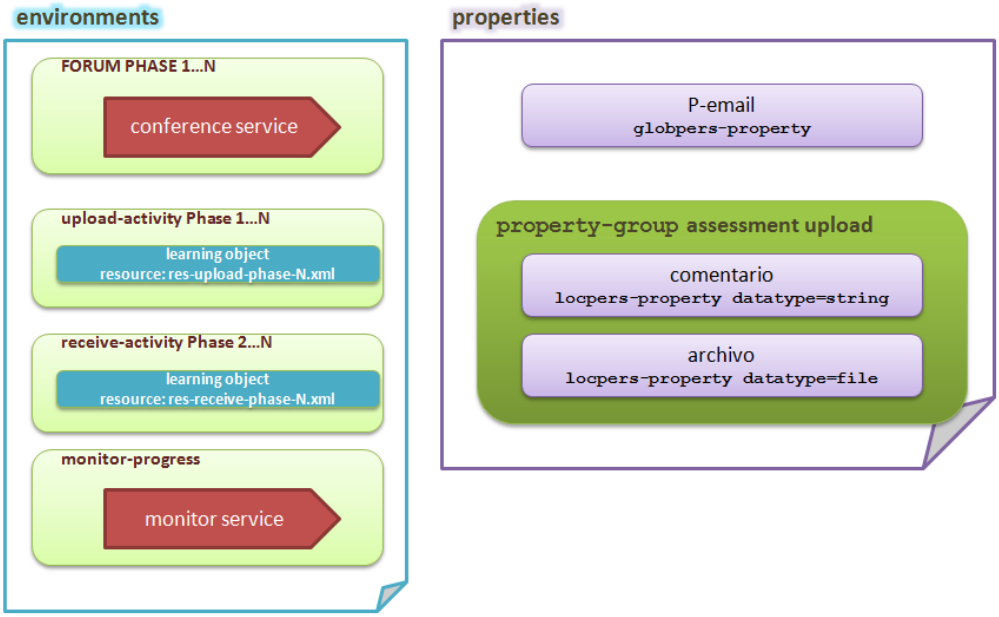


Figura 3-16. Roles y *activities* para la pedagogía orientada a proyectos en ciclo de vida

Los *environments* serán utilizados para colocar los *learning-objects* para hacer los *uploads* y poder visualizar los archivos en la anterior fase del proyecto.

En la figura anterior, se muestra un grupo de *properties* (*assessment upload*) que serán las encargadas de almacenar los comentarios y los archivos que son resultado de cada fase de desarrollo. Para un proyecto de N fases existirán por lo tanto N *assessment uploads.*

Aparte de las *properties* relacionadas con los *assesments* de los líderes de las fases, existe una propiedad *globpers-property* que será empleada para enviar las notificaciones de tipo email.

Al finalizar una fase del proyecto, la plataforma lo notifica a los integrantes y al líder de la siguiente fase para que se preparen para acometer su parte del proyecto. Esta acción se realiza mediante el empleo de la etiqueta *<imsld:notification>* del nivel C de la especificación IMS-LD. En el siguiente fragmento de código se puede ver la manera en la que se emplean estas notificaciones:

```
<imsld:learning-activity identifier="LA-END-PHASE-3" isvisible="true">
    <imsld:title>Phase: Desarrollo y pruebas has finished.</imsld:title>
    <imsld:activity-description>
            <imsld:item identifier="I-END-PHASE-3" identifierref="RES-END-PHASE-3" />
    </imsld:activity-description>
    <imsld:complete-activity>
            <imsld:when-property-value-is-set>
                    <imsld:property-ref ref="File-3" />
            </imsld:when-property-value-is-set>
    </imsld:complete-activity>
    <imsld:on-completion>
            <imsld:notification>
                    <imsld:email-data email-property-ref="P-Email">
                            <imsld:role-ref ref="ROL-80b629be-4d75-432f-9220"/>
                    </imsld:email-data>
                    <imsld:learning-activity-ref ref="LA-END-PHASE-3" />
                    <imsld:subject>
                            Phase Desarrollo y pruebas of the project has finished.
                    </imsld:subject>
            </imsld:notification>
            <imsld:notification>
                    <imsld:email-data email-property-ref="P-Email">
                            <imsld:role-ref ref="ROL-LEADER-80b629be-4d75-432f-9220" />
                    </imsld:email-data>
                    <imsld:learning-activity-ref ref="LA-END-PHASE-3" />
                    <imsld:subject>
                            Phase Desarrollo y pruebas of the project has finished.
```

```
                </imsld:subject>
            </imsld:notification>
            <imsld:notification>
                <imsld:email-data email-property-ref="P-Email">
                    <imsld:role-ref ref="PROJECT_CHIEF" />
                </imsld:email-data>
                <imsld:learning-activity-ref ref="LA-END-PHASE-3" />
                <imsld:subject>
                    Phase Desarrollo y pruebas of the project has finished.
                </imsld:subject>
            </imsld:notification>
    </imsld:on-completion>
</imsld:learning-activity>
```

El *method* para esta pedagogía va a consistir en dos *plays* diferentes. Un *play* será exclusivo para el rol de *staff* Jefe de Proyecto y el otro *play* será para el resto de los usuarios. Se puede hacer de esta manera porque cuando en IMS-LD coexisten dos *plays*, estos se ejecutan e interpretan en paralelo y de forma independiente. Se muestra en la siguiente figura el *method* para esta pedagogía:

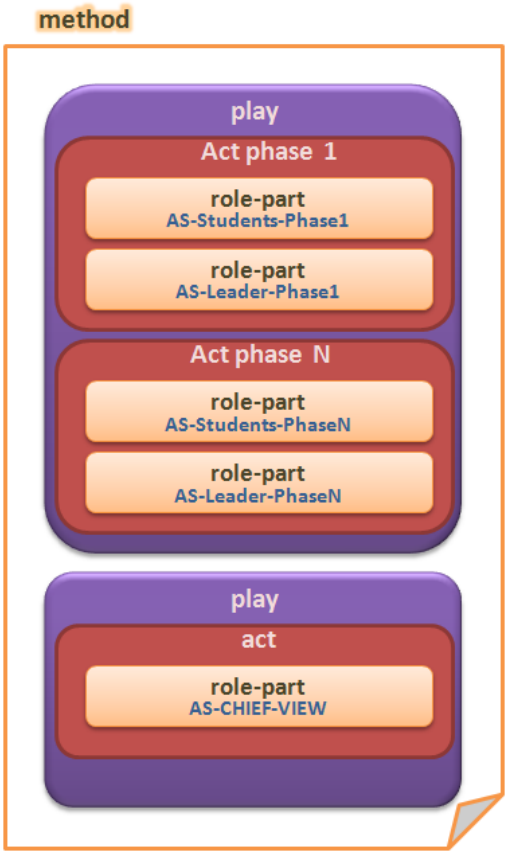


Figura 3-17. Roles y *activities* para la pedagogía orientada a proyectos en ciclo de vida

Como se muestra en la figura anterior, el *play* del jefe de proyecto (CHIEF-VIEW) contiene únicamente un *act* en el que se asigna un único *role-part* para que el rol de Jefe de Proyecto ejecute la *activity-structure* que contiene el servicio de monitorización de los *output* de las diferentes fases.

El *play* principal será en el que se encuentren todos los usuarios tipo *learner*, tanto líderes como *students.* Este *play* contendrá tantos actos como fases tenga el proyecto. La condición de finalización de cada acto, así como la inicialización del acto siguiente, vendrá determinada por la finalización del *role-part* del líder de la fase.

En el siguiente fragmento de código se muestra uno de los actos del *play* principal.

```
<imsld:act>
    <imsld:title>Project: Proyecto PLD. Phase1 : Estudio
           tecnologico</imsld:title>
    <imsld:role-part identifier="RP-c67dd2e7-1ff7-4329-94d1">
           <imsld:title>Role part learner</imsld:title>
           <imsld:role-ref ref="ROL-c67dd2e7-1ff7-4329-94d1" />
           <imsld:activity-structure-ref ref="AS-c67dd2e7-1ff7-4329-94d1" />
    </imsld:role-part>
    <imsld:role-part identifier="RP-LEADER-c67dd2e7-1ff7-4329-94d1">
           <imsld:title>Role part leader</imsld:title>
           <imsld:role-ref ref="ROL-LEADER-c67dd2e7-1ff7-4329-94d1" />
           <imsld:activity-structure-ref ref="AS-LEADER-c67dd2e7-1ff7-4329-94d1" />
    </imsld:role-part>
    <imsld:complete-act>
           <imsld:when-role-part-completed ref="RP-LEADER-c67dd2e7-1ff7-4329-94d1" />
    </imsld:complete-act>
</imsld:act>
```

Como se puede observar en el XML, el acto se considera completado cuando el líder de la fase haya completado la *activity-structure* con id: `AS-LEADER-c67dd2e7-1ff7-4329-94d1`. Para completar esta *activity-structure* el líder tendrá que subir a la plataforma el archivo de output de la fase.

4. DESARROLLO DE LA APLICACIÓN

En este apartado se abordarán los aspectos relacionados con el análisis, el diseño y la posterior implementación de la Herramienta de Autor generadora de IMS-LD.

4.1. ANÁLISIS

4.1.1. Especificación de requisitos

La aplicación a desarrollar será una aplicación web que deberá recoger datos y ficheros desde un usuario para almacenarlos y poder generar en algún momento bajo demanda archivos comprimidos de vuelta al cliente. Los datos que recogerá la aplicación desde el usuario estarán relacionados con la creación de cursos y por otro lado, los ficheros que la aplicación estarán acordes con los datos introducidos por el usuario así como a la especificación IMS-LD.

Como se ha mostrado a lo largo de este documento, un curso creado siguiendo la especificación IMS-LD se declara en archivos XML *(imsldmanifest.xml)* que no son fáciles de crear por parte de los diseñadores de cursos. La curva de aprendizaje de IMS-LD es lo suficientemente pronunciada como para sugerir la existencia de herramientas intermedias entre conceptos globales (como la orientación pedagógica) e implementación concreta (ficheros XML de mapeo).

El objetivo de esta aplicación es simplificar a los docentes la tarea de creación de cursos para que a partir de conceptos pedagógicos y siguiendo unas sencillas instrucciones, puedan generar los ficheros necesarios para poner en marcha un curso basado en IMS-LD en una plataforma compatible.

Una de las premisas para el desarrollo de esta herramienta es que sea de acceso y manejo sencillo por parte de los usuarios. Las aplicaciones web tienen la ventaja de que se ejecutan en un servidor y no requieren una instalación específica del software en el equipo cliente que accede a ella para poder ejecutarlo.

Las aplicaciones web basadas en Java proporcionan una base sólida y fiable para realizar aplicaciones escalables, flexibles y bien organizadas. El esquema general de la aplicación es el siguiente:

Figura 4-1. Esquema de la aplicación web para la implementación de la herramienta de autor

Como es recomendable para una aplicación web de estas características, se empleará un patrón de diseño modelo-vista-controlador (MVC) en el que la Vista se encargará de presentar los textos, imágenes y formularios a los usuarios, el Modelo definirá aquellos aspectos relacionados con la persistencia y el almacenaje de los distintos objetos de negocio y el Controlador se encargará de gestionar el *workflow* de la aplicación.

La lógica de negocio de la aplicación web se basa en generación de ficheros XML a partir de la información recogida desde el usuario. Estos ficheros XML (*imsmanifest.xml*), deben generarse acorde a la especificación IMS-LD y a partir de la información recogida del usuario e irán contenidos en el paquete archivo comprimido entregado al cliente junto con el resto de documentos que compondrán el curso.

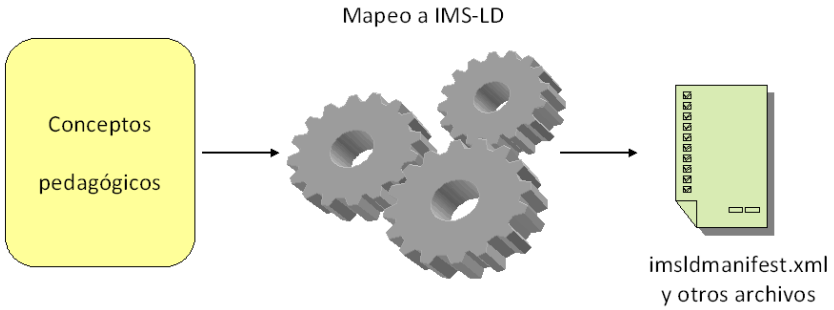


Figura 4-2. Representación gráfica del mapeo a IMS-LD que realizará la herramienta de autor

La herramienta de autor residirá en un servidor web que recibirá peticiones HTTP y responderá con contenido HTML (junto con otros recursos web CSS, JS, imágenes…) o directamente con un archivo *.zip* autocontenido resultante de la creación de un curso. Estos archivos comprimidos podrán ser cargados en una plataforma educativa compatible para su despliegue y ejecución.

El archivo comprimido debe ser por lo tanto válido para ser desplegado en un reproductor de cursos basados en la especificación IMS-LD. Por otro lado, de la anterior figura se extrae la conclusión de que el servidor en el que se despliegue la aplicación deberá almacenar los ficheros que el creador del curso envíe para así tenerlos localizados en el momento en el que se vaya a generar el archivo comprimido a enviar de vuelta al cliente.

Para la aplicación se va a considerar que todos los usuarios que accedan a la aplicación web compartirán el acceso al mismo repositorio de cursos creados. Al tratarse de una aplicación desarrollada para fines de investigación y demostración, no se van a implementar sistemas de autenticación ni autorización de los usuarios. Aunque se trataría de un aspecto a tener en cuenta como futura ampliación de la aplicación.

Adicionalmente, el servidor deberá contener un motor de base de datos que almacene los parámetros introducidos por el usuario como por ejemplo, nombre de las actividades a realizar, orden de las actividades, etc… por lo que será necesario realizar un análisis de las tablas necesarias en la base de datos.

La aplicación web soportará internacionalización de los contenidos para textos en el idioma español e inglés, este último por defecto. Para conseguirlo, los diferentes elementos de la vista no contendrán cadenas de texto explícitas sino referencias a archivos de *properties* que almacenan el listado de palabras y frases empleadas en la vista por la aplicación.

4.1.2. Diagramas de casos de uso y secuencia

El desarrollo de la aplicación se enfoca a arquitectura web para permitir el acceso de diferentes usuarios en un principio sin la necesidad de realizar *login* en la aplicación. De esta manera, todos los usuarios de la aplicación comparten un repositorio común de cursos que pueden editar y descargarse para desplegarlos en plataformas de *elearning* con soporte IMS-LD.

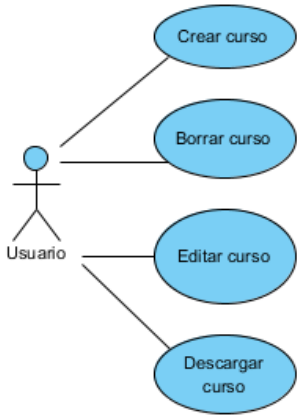


Figura 4-3. Casos de uso para la aplicación

Existe un repositorio común de cursos y materiales a los que todos los usuarios pueden acceder sin restricciones. En futuras revisiones del software, se podría implementar una asignación de creadores de cursos a sus cursos y la posibilidad de establecer diferentes perfiles de acceso para la aplicación como por ejemplo: Administrador, Creador de contenidos, Usuario de sólo lectura, etc…

4.2. DISEÑO

4.2.1. Arquitectura del sistema

La aplicación desarrollada está basada en el patrón MVC para aplicaciones web, de manera que la lógica de negocio se centra en la creación de contenidos XML que cumplan con la especificación IMS-LD. Para cumplir con estos requisitos, se han desarrollado los siguientes bloques funcionales en la aplicación.

- Capa de la vista. Desarrollado con JSP, Javascript y CSS. Se utiliza el patrón de creación de plantillas Tiles [74] para poder crear vistas que compartan un mismo *look and feel.*
- Capa de control. Se deja la parte del control de las peticiones y respuestas HTTP al *framework* de Struts 1.1.
- Lógica de negocio. Por un lado, se ha desarrollado una librería completa de objetos IMS-LD que mapean las diferentes entidades de la especificación. Adicionalmente, cada pedagogía implementa su propia manera de utilizar los objetos IMS-LD para satisfacer los requisitos de la orientación pedagógica en función de los *inputs* introducidos por el cliente y guardados en la base de datos.
- Base de datos. El motor de base datos utilizado es MySQL [75] y el *framework* con el que se accede a los datos es Hibernate [78]. El patrón de diseño aplicado para el acceso a los datos guardados en la base de datos es DAO.

En la siguiente figura se muestran los diferentes elementos que componen la arquitectura de la aplicación desarrollada para este proyecto.

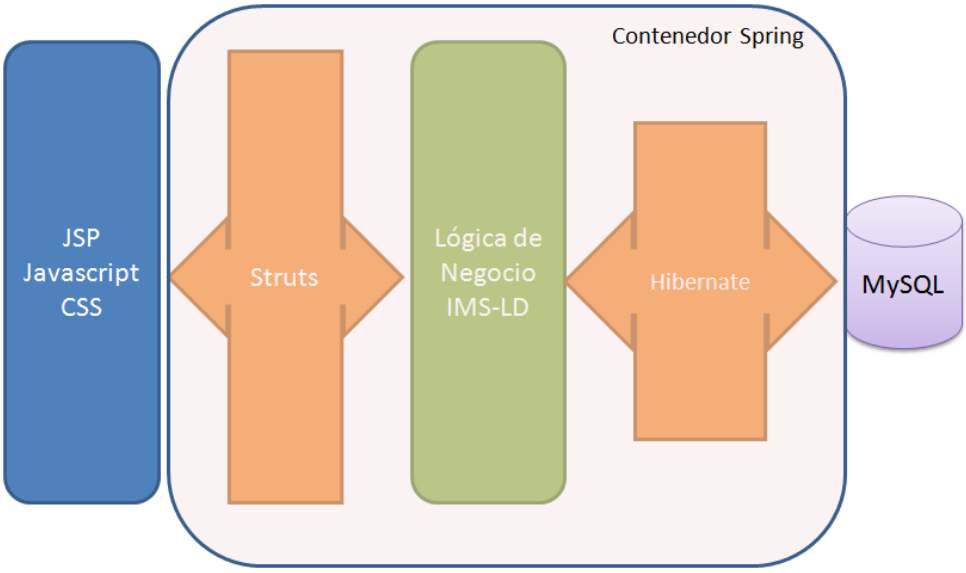


Figura 4-4. Arquitectura software de la aplicación web desarrollada

Spring [79] es un *framework* muy completo para desarrollo de aplicaciones (principalmente en Java) que se emplea en el proyecto para aportar flexibilidad a la aplicación. Concretamente, se emplean las características de inyección de dependencias para proporcionar independencia entre las diferentes capas de las que consta la aplicación. El detalle de la arquitectura software empleada se muestra en detalle en el apartado 4.2.3.

4.2.2. Modelo de datos

En todas las pedagogías implementadas se sigue una misma forma de diseño para las tablas. Básicamente, todas las pedagogías tienen en común los datos del curso que se está realizando y el conjunto de recursos que componen el curso (rutas a ficheros en el servidor o URL para servir el contenido).

A continuación se presentan los esquemas de base de datos utilizados para la implementación de cada pedagogía.

4.2.2.1. Pedagogía lineal

La pedagogía lineal únicamente agrupa para cada curso un conjunto de actividades. Por lo que en este caso, solamente están involucradas la tabla de cursos y la de actividades de cursos de pedagogía lineal.

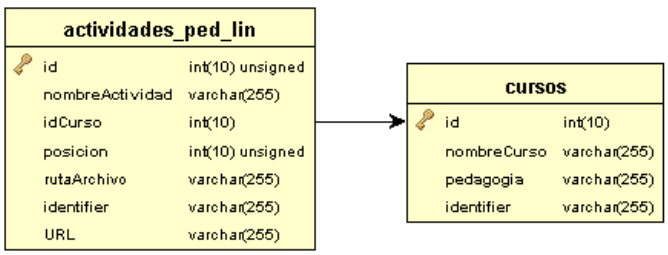


Figura 4-5. Esquema BD para pedagogía lineal

4.2.2.2. Pedagogía cognitivista

La pedagogía cognitivista asocia para cada curso un conjunto de bloques que se van mostrando o repitiendo en función de ciertas condiciones. Los recursos para esta pedagogía son almacenados en la tabla de *resources_ped_connectionist* y los umbrales elegidos para que el alumno progrese hasta el siguiente curso se almacenan en la tabla *marks_ped_connectionist*.

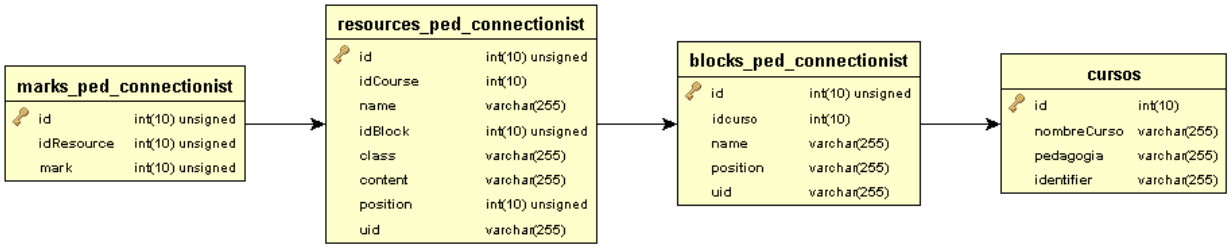


Figura 4-6. Esquema BD para pedagogía cognitivista

4.2.2.3. Aprendizaje basado en los cuatro pasos de Polya

La pedagogía basada en los cuatro pasos de Polya agrupa un conjunto de problemas para cada curso. La tabla de *problems_ped_polya* almacenará los datos relacionados con cada problema y sus recursos asociados se almacenan en *resources_ped_polya.*

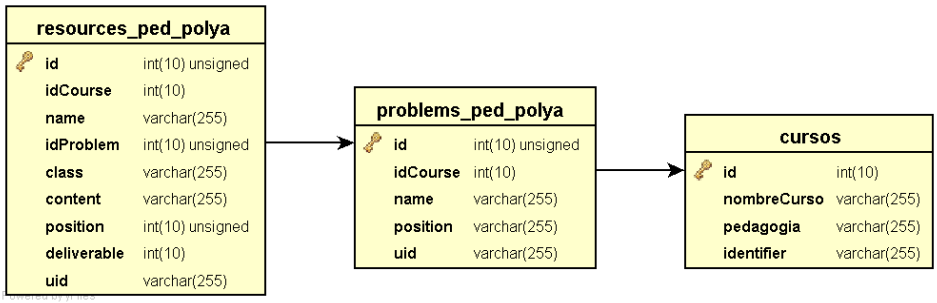


Figura 4-7. Esquema BD para pedagogía 4 pasos de Polya

4.2.2.4. Aprendizaje basado en los tres pasos de Mason, Burton y Stacey

La pedagogía basada en los tres pasos de Mason, Burton y Stacey es un subconjunto de la pedagogía de los cuatro pasos de Polya por lo que al igual que en el anterior punto, la tabla de *problems_ped_mason* almacenará los datos de los problemas para cada curso y los recursos asociados se almacenarán en *resources_ped_mason.*

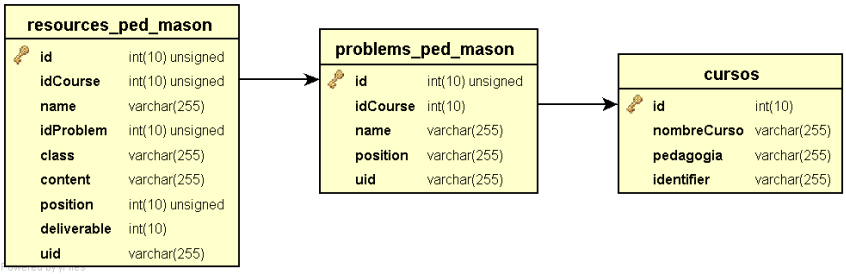


Figura 4-8. Esquema BD para pedagogía 3 pasos de Mason Burton y Stacey

4.2.2.5. Aprendizaje basado en problemas para ingeniería

Para el diseño de estas tablas se ha tenido en cuenta que cada curso contiene uno o varios problemas que se almacenan en la tabla *problems_ped_pbl* y un conjunto de recursos que se almacenan en *resources_ped_pbl*.

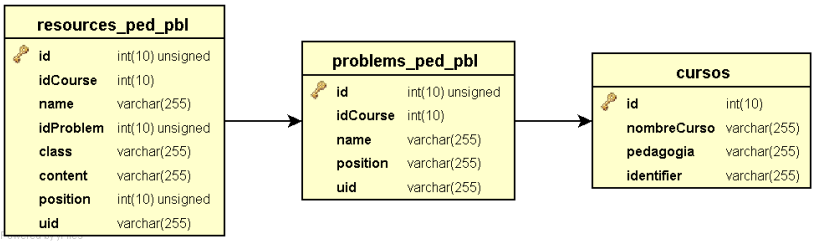


Figura 4-9. Esquema BD para pedagogía PBL

4.2.2.6. Aprendizaje basado en proyectos con roles paralelos

Este tipo de cursos consta de un conjunto de partes que a su vez está compuesta por un conjunto de subpartes. Como en todas las pedagogías, los recursos necesarios se almacenan en la tabla de recursos *resources_ped_prosim* y se relacionan con cada subparte a través de la clave extranjera *idSubpart*.

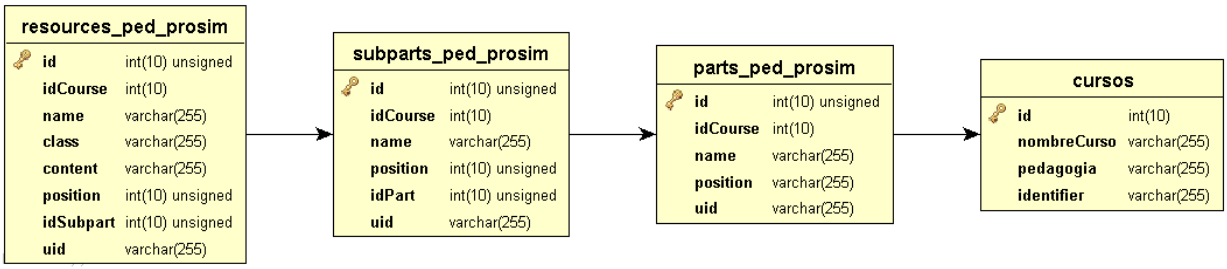


Figura 4-10. Esquema BD para proyectos con roles paralelos

4.2.2.7. Aprendizaje basado en proyectos en ciclo de vida

Pese a que este tipo de cursos es muy diferente en términos de funcionalidad al aprendizaje basado en roles paralelos, el diseño es una simplificación del anterior ya que cada curso solamente contiene un conjunto de partes que se guardan en *parts_ped_prolife*. Los recursos se almacenan en la tabla de *resources_ped_profile* asociados con las partes a través del campo *idPart*.

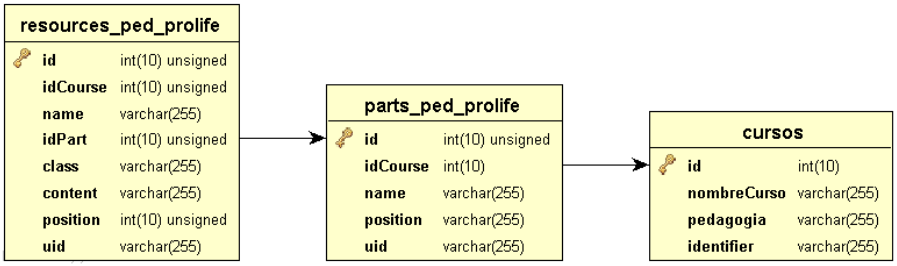


Figura 4-11. Esquema BD para proyectos en ciclo de vida

4.2.2.8. Modelo de datos completo

El esquema de la base de datos que utiliza la aplicación es el siguiente:

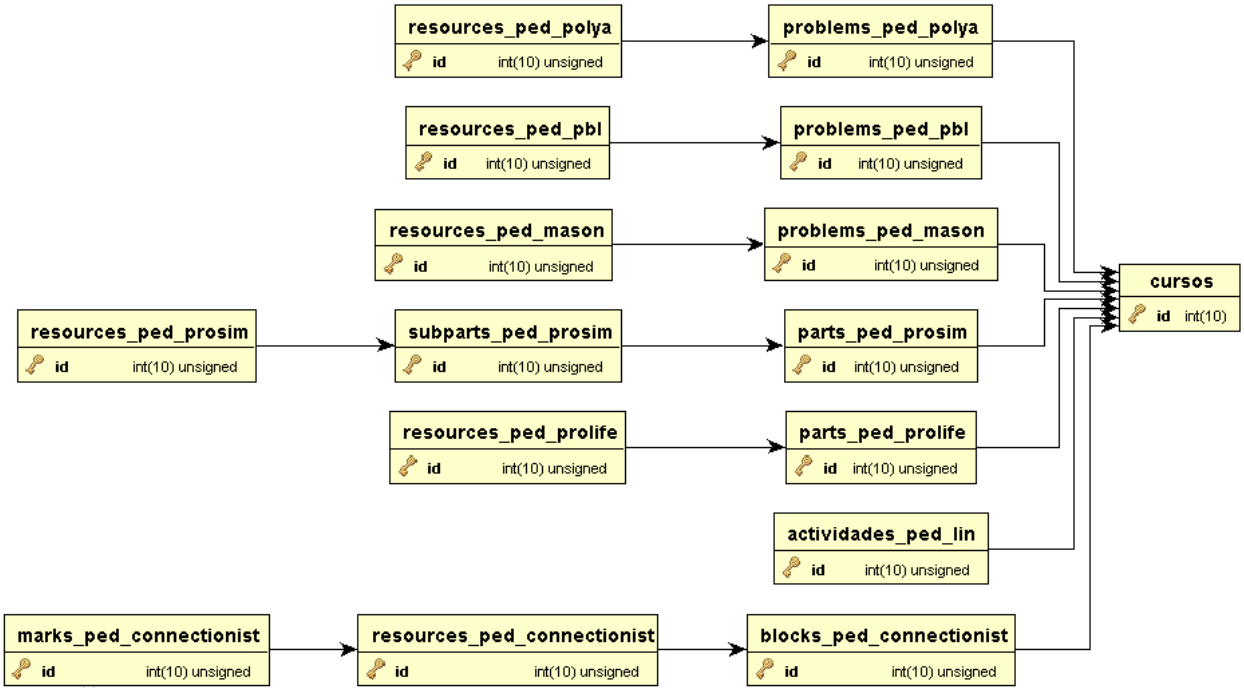


Figura 4-12. Esquema de la base de datos de la aplicación web

4.2.3. Arquitectura software

Como en la mayoría de los proyectos software, la parte diferenciada de la herramienta es la lógica de negocio. En este caso, la lógica de negocio se centra en el mapeo a términos IMS-LD de la información recogida desde el usuario para generar el descriptor principal (*imsldmanifest.xml*) además, algunas pedagogías requerirán que se generen archivos adicionales a los archivos de recursos que incluya el docente. Por ejemplo en el caso de las pedagogías que requieren una monitorización por parte del docente, se crearán además del *manifest* los archivos necesarios monitor.xml para proporcionar este tipo de operaciones.

Para poder realizar el mapeo a IMS-LD se deben conocer las características de los diferentes conceptos de la especificación y su modelo de información. En la siguiente figura, se muestra la arquitectura empleada para el desarrollo de la aplicación:

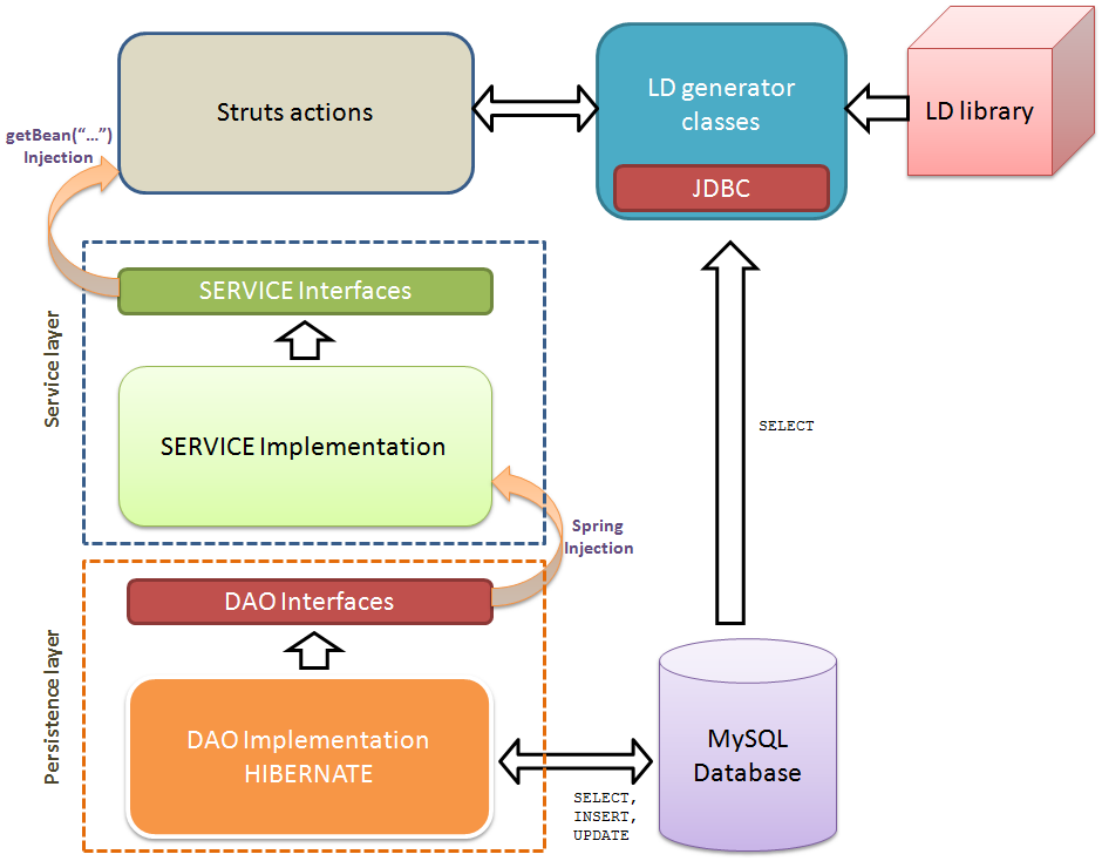


Figura 4-13. Arquitectura de la aplicación

El eje central que gestiona las peticiones y respuestas a los clientes web es Struts quien en función del tipo de petición solicitada, deriva la ejecución hacia las clases de servicio para hacer lecturas y escrituras de la base de datos o bien deriva la ejecución hacia las clases generadoras de IMS-LD.

Por cada pedagogía se ha implementado una *action* de Struts (`EditXxxxAction`) y un generador de LD (`EditXxxxBean`).

Cuando un usuario decide descargar el archivo *.zip* que contiene el curso, las acciones de Struts ejecutan el siguiente código:

```
if (action.equals("generate")) {
        // ************************************************
        // GENERATE DEL CURSO COMPLETO
        // ************************************************
        EditPblBean bean = new EditPblBean();
        // Path local donde se encuentran los archivos del curso
        String path = Common.getPath(curso);
        bean.setPath(path);
        bean.setCurso(curso);
        bean.doManifest();
        bean.createFile();
        Common.generateZIP(curso);
        Common.sendToClient(curso , response);

        //La respuesta ya se ha llevado a cabo enviando el archivo comprimido
        //por eso se devuelve un forward null
        return null;

}
```

Por lo tanto, la creación del *manifest* comienza con la llamada a `bean.doManifest()`. Las clases generadoras de IMS-LD realizan su propio mapeo, convirtiendo los resultados obtenidos de las consultas SQL contra la base de datos en objetos LD mediante el uso de la librería de LD desarrollada en este proyecto. Más adelante, en el apartado 4.2.5 se profundizará sobre la librería LD y la generación del *manifest*.

En el siguiente fragmento de código se observa la utilización de los objetos de la librería de LD por parte de las clases generadoras de LD. El código pertenece a la clase `EditPblBean.java`:

```
public Method makeMethod(Act act){
        //Title Play
        Title title_play = new Title(LD_NS);
        title_play.setTitle("Course: "+curso.getNombreCurso());
        //Play
        Play play = new Play (LD_NS);
        play.setTitle(title_play);
        play.setAct(act);
        //Method
        Method method = new Method(LD_NS);
        method.setPlay(play);
        method.setConditions(conditionsTag);

        return method;
}
```

Como se muestra en la figura de arquitectura de la aplicación, se han empleado dos vías de acceso a los contenidos de la base de datos MySQL. Por un lado, las clases generadoras de código IMS-LD hacen llamadas directas a la base de datos mediante el empleo de conectores directos JDBC [77] mientras que las acciones de Struts, hacen llamadas a la base de datos a través de un conjunto de clases DAO de Hibernate.

Las clases generadoras de IMS-LD únicamente realizan operaciones de lectura SELECT para recuperar el contenido de las tablas que se requieren para generar el *manifest* de LD mientras que los DAOs de Hibernate realizan las operaciones de SELECT, INSERT y UPDATE para recuperar e introducir datos en las tablas de la base de datos a medida que el usuario de la aplicación va realizando el diseño del curso a través de las diferentes pantallas.

El motivo por el que no se utiliza Hibernate en las clases generadoras de IMS-LD es porque estas clases aglutinan el completo de la lógica de negocio de la aplicación y el hecho de integrar conectores JDBC simplifica la lectura del código ya que Hibernate emplea un lenguaje propietario de acceso a datos HQL y en ocasiones consultas embebidas. Al ser éste un proyecto con fines académicos, se ha pretendido implementar la parte más diferenciada de la aplicación de la manera más sencilla para la lectura del código sin necesidad de indagar en clases de Servicio y de Hibernate.

Las clases de Struts y las clases de Servicio están aisladas por una capa de interfaces (*SERVICE Interfaces*). Igualmente, la capa de servicio *SERVICE LAYER* y la capa de persistencia *PERSISTENCE LAYER* están igualmente aisladas por la capa de interfaces *DAO Interfaces*. Esta forma de trabajar permite que en un momento dado, las implementaciones puedan ser sustituidas por otra tecnología sin necesidad de modificar el código de las capas superiores. Por ejemplo, sería posible migrar la aplicación a otro tipo de mapeo ORM (iBatis) o directamente una capa de conectores JDBC en lugar de Hibernate.

Para lograr el objetivo de interdependencia entre capas es para lo que se utiliza Spring en este proyecto. La particularidad empleada es la inyección de dependencias mediante inversión de control, en el apartado 4.3.5 de este documento se profundiza sobre estos aspectos de Spring.

En las *actions* de Struts, las instancias de las clases de servicio se obtienen de la siguiente manera:

```
public class EditPblAction extends org.apache.struts.action.Action {

     private ProblemsPedPblServicio problemsPedPblSrv;
     public ActionForward execute( . . . ) throws Exception {
            problemsPedPblSrv =
                  (ProblemsPedPblServicio)Common.getBean("problemsPedPblServicio", session);
                    // Código de la action de Struts
```

Spring habrá definido el bean llamado `problemsPedPblServicio` indicando a qué clase pertenece y sus propiedades de instanciación. En el archivo `applicationContext-servicios.xml` se declara el bean:

```
     <bean id="problemsPedPblServicio" class="src.servicios.ProblemsPedPblServicioImpl">
       <property name="problemsPedPblDAO" ><ref bean="problemsPedPblDAO"/></property>
     </bean>
```

Una de las propiedades del bean: `problemsPedPblDAO` está gestionada por Spring y es el propio *framework* Spring quien utiliza el método `set()` del bean para insertar el objeto DAO durante el tiempo de ejecución. En el archivo `applicationContext-dao.xml` se declara el bean `problemsPedPblDAO`:

```
     <bean id="problemsPedPblDAO" class="src.dao.hibernate.ProblemsPedPblDAOImpl">
            <property name="sessionFactory" ref="hibernate" />
     </bean>
```

Como se puede ver en el fragmento de código anterior, los DAOs reciben el objeto de sessionFactory de Hibernate para establecer la comunicación con la base de datos dentro de un marco gestionado por Spring. La única dependencia en el código del bean de servicio para poder insertar un objeto de la capa DAO es el método `set()`:

```
public class ProblemsPedPblServicioImpl implements ProblemsPedPblServicio {

     private ProblemsPedPblDAO problemsPedPblDAO;

     public void setProblemsPedPblDAO(ProblemsPedPblDAO problemsPedPblDAO) {
            this.problemsPedPblDAO = problemsPedPblDAO;
     }
     //Método de ejemplo de uso del DAO
     public List<ProblemsPedPbl> getAllProblems(Curso curso) {
            return problemsPedPblDAO.getAllProblems(curso.getIdentifier());
     }
```

La clave de esta explicación está en que la clase de implementación del servicio `ProblemsPedPblServicioImpl` no está acoplada a la capa de implementación de los DAOs sino a la capa de interfaces DAO, consiguiéndose así una interdependencia real entre las capas de la aplicación.

El proceso mediante el cual un bean de la capa de servicio obtiene un bean de la capa DAO es lo que se llama en la figura anterior *Spring injection.*

Las acciones de Struts sin embargo, no utilizan este mecanismo para obtener los beans sino que utilizan un método estático de la clase `Common.java`: `getBean()` que recupera del contexto de Spring el bean requerido por el String que recibe por parámetro:

```
     public static Object getBean(String bean, HttpSession session) {
            ApplicationContext applicationContext =  WebApplicationContextUtils.
                                    getWebApplicationContext(session.getServletContext());
            return applicationContext.getBean(bean);
     }
```

El motivo por el que las acciones de Struts no recuperan los objetos mediante IoC de Spring es porque no se han definido como beans de Spring. Aunque técnicamente es viable dar de alta las acciones de Struts como beans de Spring, la configuración de Struts se vuelve más opaca de cara al desarrollador, por lo que las acciones de Struts se han decidido mantener fuera del control de Spring.

4.2.4. Diseño de clases

La herramienta de autor generadora de IMS-LD es una aplicación web que cumple con la estructura convencional de carpetas para este tipo de aplicaciones en Java:

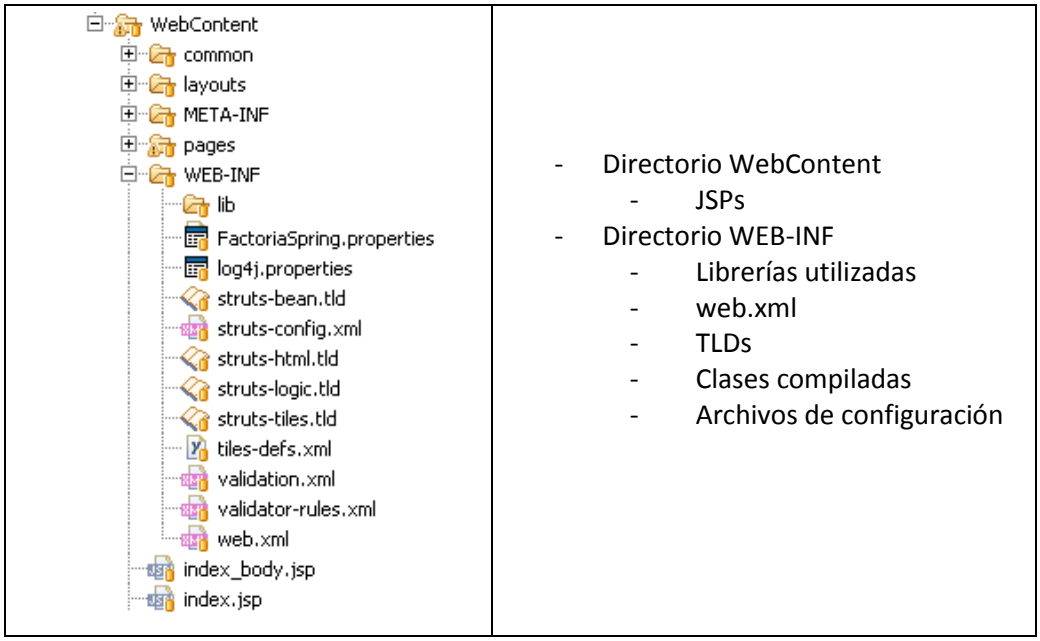


Figura 4-14. Archivos de la aplicación web

En la siguiente figura se muestra todos los paquetes Java implementados para el desarrollo de este proyecto. Como se puede observar, por cada pedagogía se ha implementado un paquete de Java:

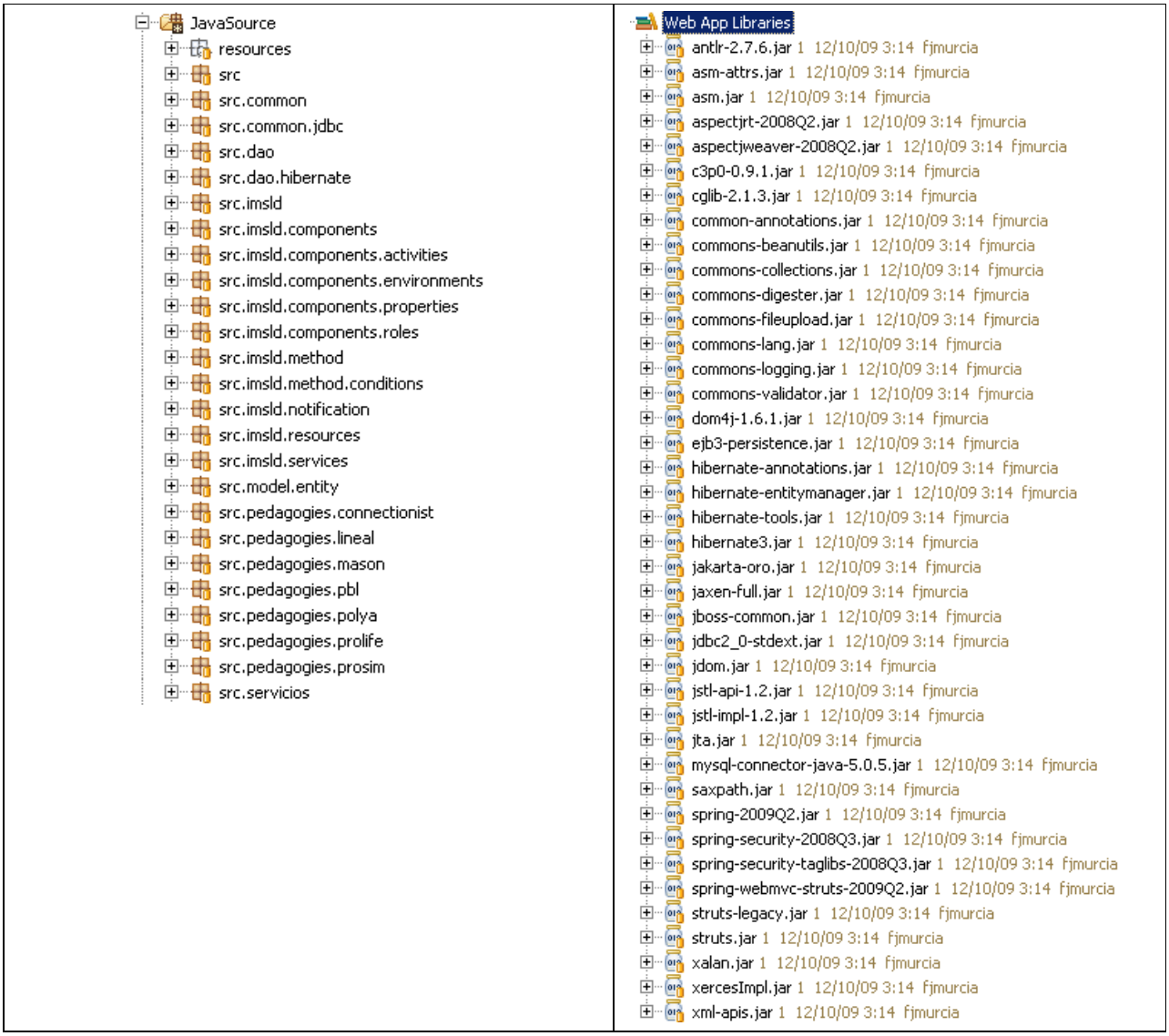


Figura 4-15. Paquetes Java y librerías utilizadas en la aplicación

resources	Contiene los archivos de `properties` necesarios para la internacionalización
src.common.*	Contiene las clases de utilidades utilizadas en varios puntos de la aplicación
src.imsld.*	Librería a LD
src.model.entity	Entidades de de mapeo ORM.
src.pedagogies.*	Contienen las clases de `Action`, `ActionForm` de Struts y la clase *Bean* generadora de LD.
src.servicios	Contiene las clases de servicio de la capa Service Layer de la aplicación.

Entre las librerías utilizadas por la aplicación, destacan las de Struts, Spring, Hibernate y DOM, esta última para la generación de código XML.

Uno de los aspectos técnicos más interesantes de la aplicación es la interdependencia entre capas que se consigue mediante el correcto diseño de las capas y la utilización de Spring. En la siguiente figura se muestra la relación entre las clases de DAO y sus respectivas interfaces. Como se puede observar, el DAOGeneral contiene los métodos más comunes de acceso a datos: `consultar`, `insertar`, `actualizar` y `eliminar`. El DAO de la pedagogía lineal hereda del DAOGeneral y añade los métodos particulares de esta pedagogía: `getAllActivities`, `getActividadByPosition`, `getActividadByIdentifier`.

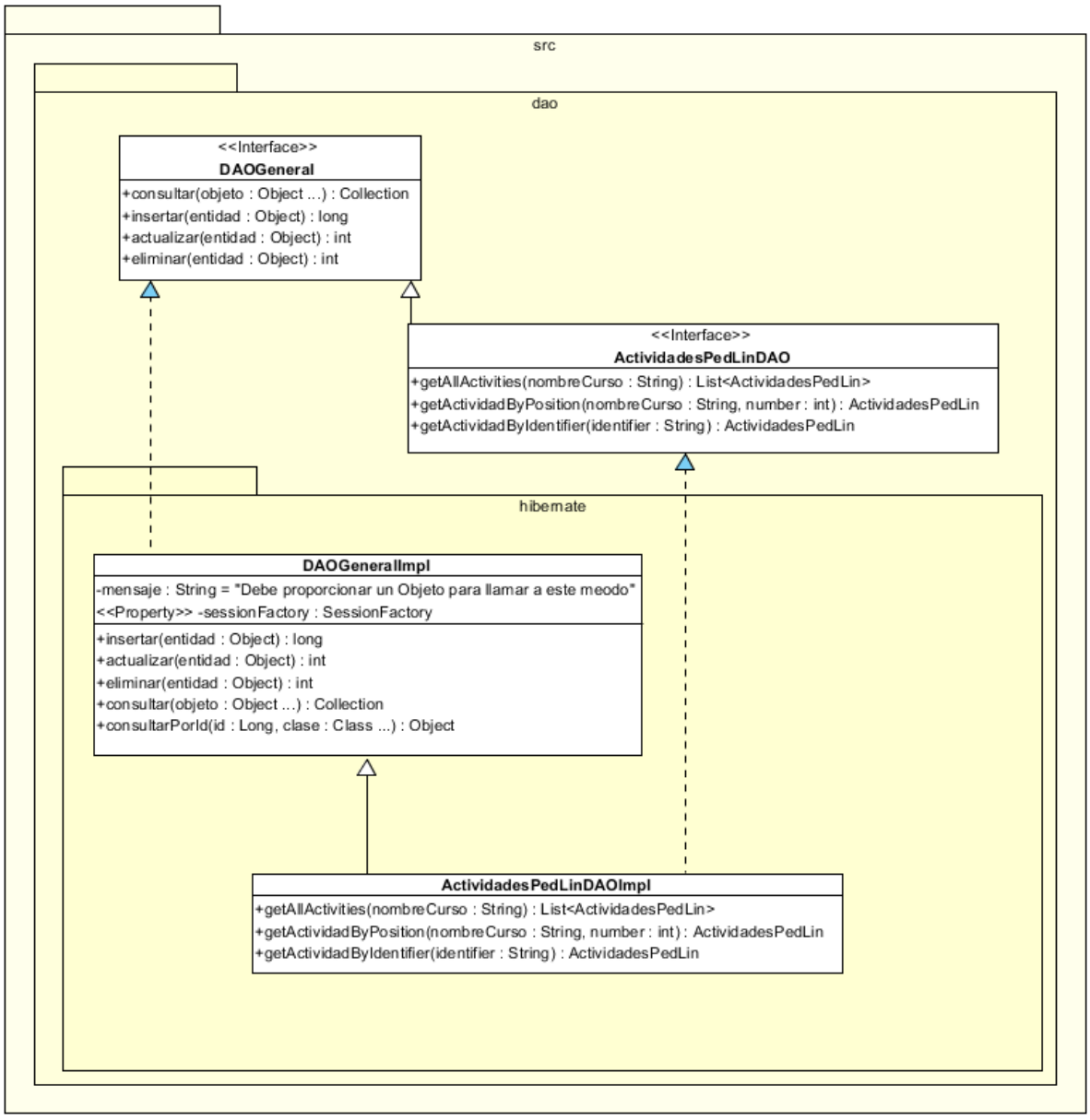


Figura 4-16. Relación entre capa de interfaces e implementación DAO

4.2.5. La librería LD y generación de *manifest*

Otro de los bloques principales de trabajo de este proyecto es el desarrollo de una librería de LD para completar el mapeo de las pedagogías. La siguiente figura recoge un diagrama de clases centrado en la clase `Learning-Design` en el que se puede observar las relaciones de uso existente entre las diferentes clases de la librería:

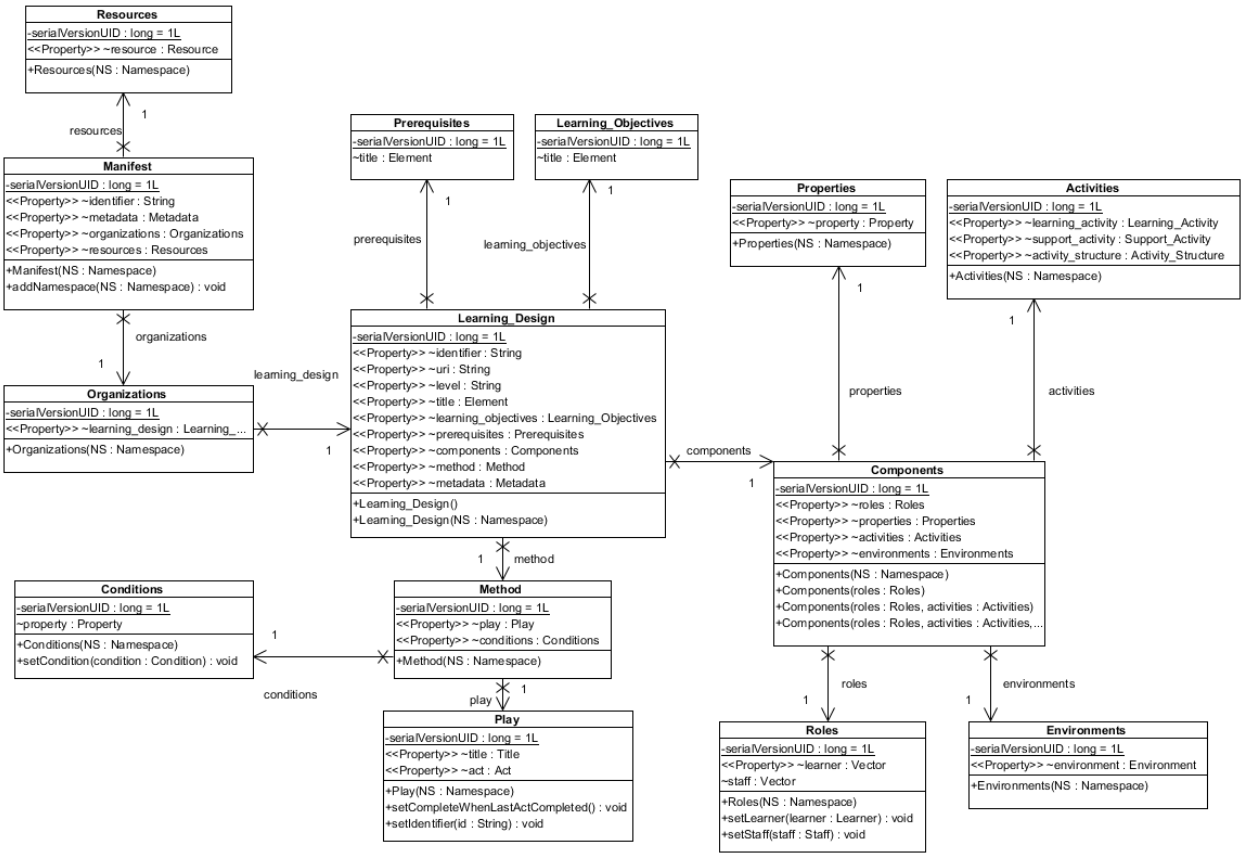


Figura 4-17. Diagrama de clases de la librería LD relacionado con el objeto *Learning-Design*

El punto clave de esta implementación es que todos los objetos de mapeo a entidades LD heredan de la clase de la libraría JDOM [82] `org.jdom.Element` por lo que es en esta librería en la que se realiza el *binding* a XML. Cuando un generador de LD utiliza la librería, lo hace de un modo similar al siguiente:

```
Manifest manifest = new Manifest(CP_NS);
manifest.addNamespace(XSI_NS);
Utils.insertarComentarios(manifest,curso.getNombreCurso());
manifest.setIdentifier("COURSE-PBL-PEDAGOGY-"+learning_design.getIdentifier());
manifest.setOrganizations(organizations);
manifest.setResources(resourcesTag);

myDoc = new Document(manifest);
```

Siendo:

```
Namespace LD_NS = Namespace.getNamespace("imsld","http://www.imsglobal.org/xsd/imsld_v1p0");
Namespace CP_NS = Namespace.getNamespace("http://www.imsglobal.org/xsd/imscp_v1p1");
Namespace XSI_NS = Namespace.getNamespace("xsi","http://www.w3.org/2001/XMLSchema-instance");
```

El constructor del objeto `org.jdom.Document` crea un documento XML cuyas etiquetas están anidadas como lo estén los objetos tipo `Element` que se le pasa como argumento. Como el objeto `Manifest` contiene un objeto `Organizations` y un objeto `Resources`, el XML generado tendrá la estructura:

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest   xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            identifier="COURSE-CONECTIONIST-PEDAGOGY-Microondas">
    <!--Microondas. File automatically generated. Created: 08 feb 2011 20:56:01 +0100-->
    <organizations>
    < .... >
  </organizations>
  <resources>
    < .... >
  </resources>
</manifest>
```

De la misma manera, siguiendo con la explicación, el objeto `Organizations` que se compone en la librería de una propiedad `Learning-Design` que a su vez se compone (entre otras) de las propiedades: `Title`, `Components` y `Method`, tendrá un mapeo en XML:

```
  <organizations>
    <imsld:learning-design>
      <imsld:title />
      <imsld:components />
      <imsld:method />
    </imsld:learning-design>
  </organizations>
```

Para concluir con este apartado, los `Components` tienen las propiedades `Roles`, `Properties`, `Activities` y `Environments` y su mapeo en XML será:

```
      <imsld:components>
        <imsld:roles />
        <imsld:properties />
        <imsld:activities />
        <imsld:environments />
      </imsld:components>
```

La creación de la librería se apoya por lo tanto en este concepto de ir creando objetos con propiedades anidadas de otros objetos como el *binding* a XML de IMS-LD recomienda [61].

Para finalizar con este apartado se muestra en el siguiente diagrama de clases la relación del objeto `Activities` con los objetos `Learning-Activity`, `Support-Activity`, `Activity-Structure`...

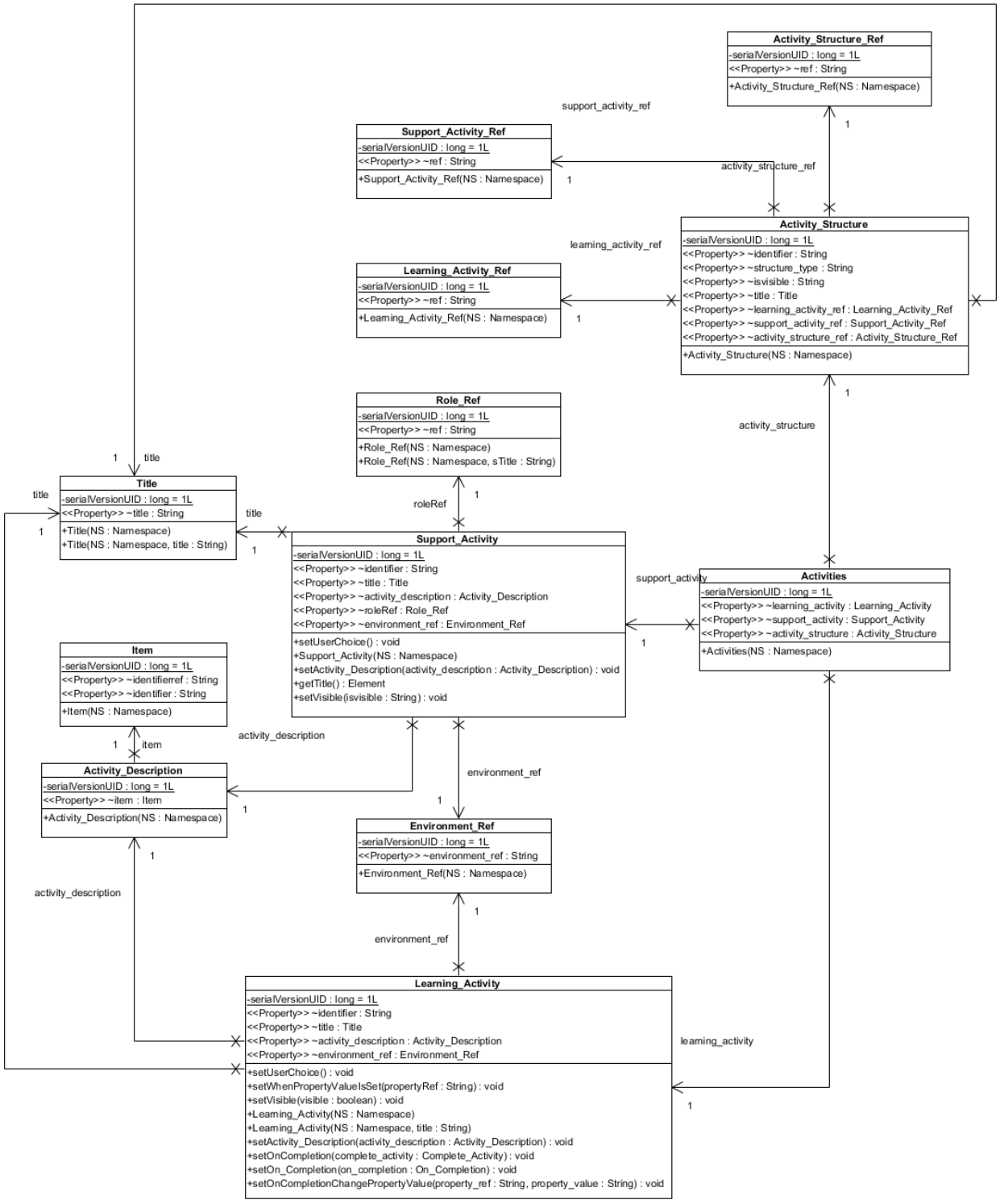


Figura 4-18. Diagrama de clases de la librería LD relacionado con el objeto *Activities*

Esta forma de trabajar con el mapeo de la especificación de IMS-LD facilita enormemente el trabajo ya que desde las clases generadoras de LD únicamente habrá que preocuparse de recuperar la información almacenada en las tablas de la base de datos y adaptarla a objetos de la librería de LD. Una vez

completado el mapeo a objetos LD, la creación del archivo de manifest XML es una simple llamada al constructor de la clase `org.jdom.Document`.

El diagrama de secuencia que sigue la aplicación para generar el *manifest* y el archivo *.zip* que contiene el curso creado se muestra en la siguiente figura particularizado para un curso creado de la pedagogía *Problem Based Learning*:

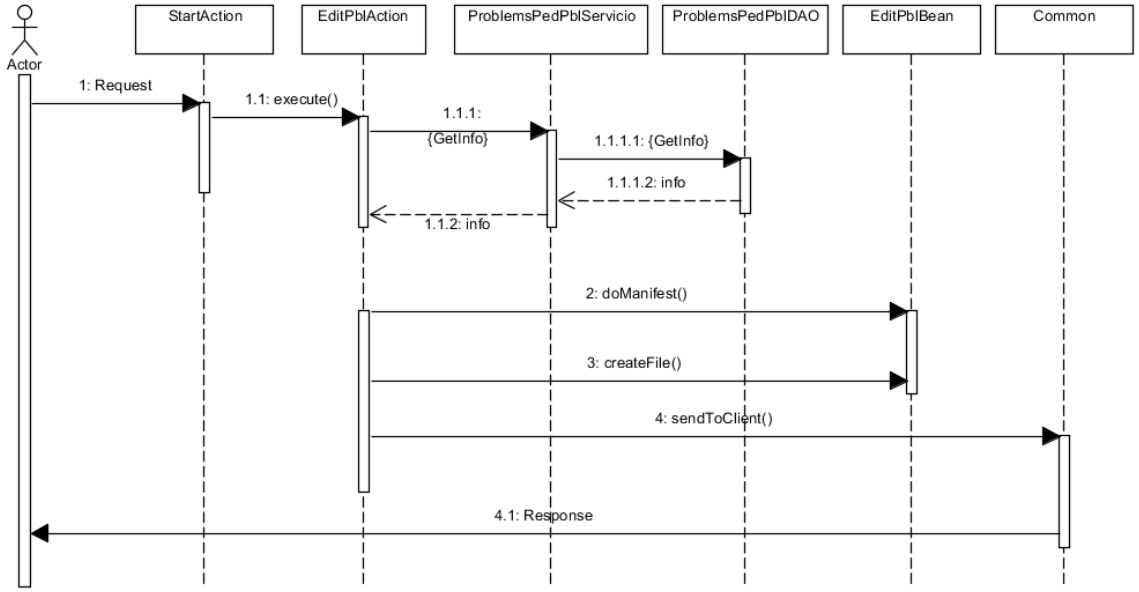


Figura 4-19. Diagrama de secuencia para descargar un curso PBL

4.3. IMPLEMENTACIÓN

4.3.1. Resumen de *frameworks* empleados para el desarrollo

En la siguiente tabla se muestra un resumen de los *frameworks* y tecnologías utilizadas para la implementación de la herramienta de autor.

Capa de negocio:	*Spring 2.0*
Capa del controlador:	*Struts 1.2, Struts Validator*
Capa de persistencia:	*Hiberante Annotations 3.0.1*
Capa de presentación:	*JSP, JSTL, Tiles, CSS, Javascript, jQuery*
Motor de base datos:	*MySQL 5.0*
Parseo XML:	*JDOM*

4.3.2. Struts 1.2

La decisión de partida a la hora de realizar la aplicación es que la gestión de las diferentes pantallas que deben ir mostrándose a los usuarios durante la navegación debe ser lo más sencilla y organizada posible. Para este objetivo, el modelo 2 del patrón MVC se ajusta perfectamente:

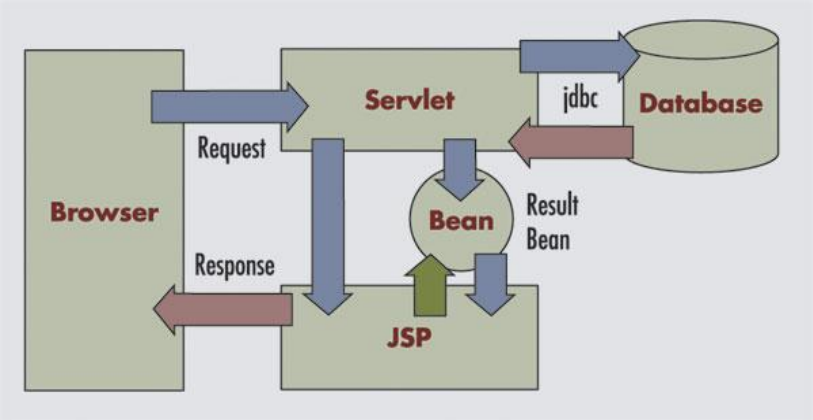


Figura 4-20. Modelo 2 del patrón de diseño MVC [83]

En la figura anterior, el *servlet* central es el Controlador. Será el encargado de recibir las peticiones HTTP, procesar la URL solicitada por el cliente y delegar el control a la clase de Java que corresponda.

El Modelo presenta dos capas que son la capa de Modelo de Acción y Modelo de Estado. La tarea del modelo se divide respectivamente en invocar cierta lógica de negocio y por otro lado en proporcionar un soporte para almacenar información a lo largo de las peticiones HTTP como por ejemplo beans de Java utilizados como Data Transfer Object almacenados en la *request* o en la *session* para trasladar a la Vista el resultado de la ejecución en la capa de acción del Modelo.

La Vista será un conjunto de Java Server Pages (JSP) acompañados de una serie de recursos que completarán las tecnologías del lado del cliente: JavaScript, hojas de estilo CSS. Podemos simplificar que la misión de la Vista es mostrar la información que reside en la capa del Modelo de Estado.

Los *frameworks* más ampliamente utilizados para implementar este modelo son *Java Server Faces* y *Struts* [84] en sus versiones 1 y 2, aunque curiosamente Struts 2 no es una evolución del código de Struts 1 sino que es una adaptación de un *framework* distinto llamado WebWork.

Comparado con los otros *frameworks*, Struts 1 presenta una forma de trabajar sencilla si bien un tanto repetitiva en ocasiones basada en la creación de clases especiales de Struts (*ActionForms*) para los distintos formularios web que se van presentando al usuario. Gracias en parte a este aspecto, Struts 1 incluye un potente y flexible marco de validación de datos de formularios tanto en el lado del cliente como en el lado del servidor: *Struts Validator*. Este *framework* de validación nos permitirá añadir controles en campos de los formularios para evitar entradas erróneas por parte del usuario, como por ejemplo, dejar un campo vacío o una URL mal formada.

Una aplicación desarrollada en Struts 1 se compone principalmente de los siguientes elementos:

- Archivo de configuración *struts-config.xml*. En este archivo se declaran los distintos formularios que existen a lo largo de la aplicación y se definen las distintas reglas de navegación disponibles.
- Clases de Action. El servlet principal de Struts delega la ejecución en las clases de Action pertinentes según el mapeo que exista en *struts-config.xml*. El método execute es el método principal que se ejecuta de estas Action.
- Clases de ActionForm. Son beans de Java capaces de almacenar de forma automática los datos de un formulario recibido desde un cliente. Los ActionForm permiten hacer validaciones internas de sus propiedades antes incluso de que el control llegue al método execute de la Action correspondiente.
- JSP con etiquetas propias de Struts. Aunque no es imprescindible utilizarlas, el empleo de etiquetas de Struts permite aprovechar realmente la potencia del *framework* de cara a introducir formularios, realizar validaciones, etc…

En la siguiente figura se muestra el proceso de recepción de una HTTP request por parte del servidor web cuando está utilizando Struts.

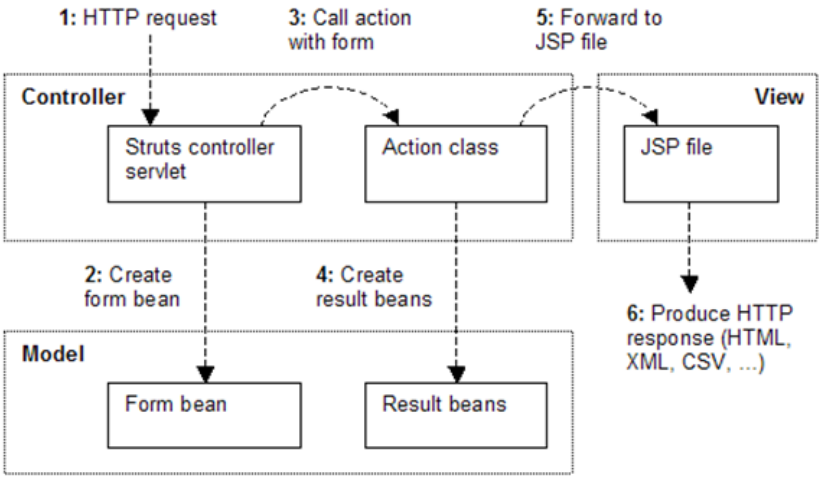


Figura 4-21. Flujo de una petición-respuesta web con Struts 1.x

El siguiente fragmento de código muestra la declaración de los distintos *ActionForm* realizada en el *struts-config.xml* de la aplicación.

```
<form-beans>
  <form-bean name="StartForm" type="src.StartForm"/>
  <form-bean name="CreateCourseForm" type="src.CreateCourseForm"/>
  <form-bean name="EditLinealForm" type="src.pedagogies.lineal.EditLinealForm"/>
  <form-bean name="EditConnectionistForm"
             type="src.pedagogies.connectionist.EditConnectionistForm"/>
  <form-bean name="EditPolyaForm" type="src.pedagogies.polya.EditPolyaForm"/>
  <form-bean name="EditMasonForm" type="src.pedagogies.mason.EditMasonForm"/>
  <form-bean name="EditPblForm" type="src.pedagogies.pbl.EditPblForm"/>
  <form-bean name="EditProsimForm" type="src.pedagogies.prosim.EditProsimForm"/>
  <form-bean name="EditProlifeForm" type="src.pedagogies.prolife.EditProlifeForm"/>
</form-beans>
```

El siguiente fragmento de código muestra la declaración de una de las acciones (*Start.do*) dentro del *struts-config.xml* de la aplicación.

```
  <action name="StartForm" path="/Start" scope="request" type="src.StartAction">
   <forward name="lineal" path="/pages/lineal1.jsp"/>
   <forward name="connectionist" path="/pages/connectionist1.jsp"/>
   <forward name="polya" path="/pages/polya1.jsp"/>
   <forward name="mason" path="/pages/mason1.jsp"/>
   <forward name="pbl" path="/pages/pbl1.jsp"/>
   <forward name="prosim" path="/pages/prosim.jsp"/>
   <forward name="prolife" path="/pages/prolife.jsp"/>
   <forward name="restoreDB" path="/pages/restoreDB.jsp"/>
   <forward name="select_course" path="/pages/select_course.jsp"/>
   <forward name="edit_course" path="/pages/edit_course.jsp"/>
   <forward name="new_course" path="/pages/new_course.jsp"/>
   <forward name="error" path="/pages/error.jsp"/>
  </action>
```

4.3.3. Tiles

Tiles [74] en un sistema de creación de plantillas. Es útil para proporcionar a las aplicaciones web un feel and look común a todas las pantallas de la aplicación y crear componentes de la vista fácimente reutilizables.

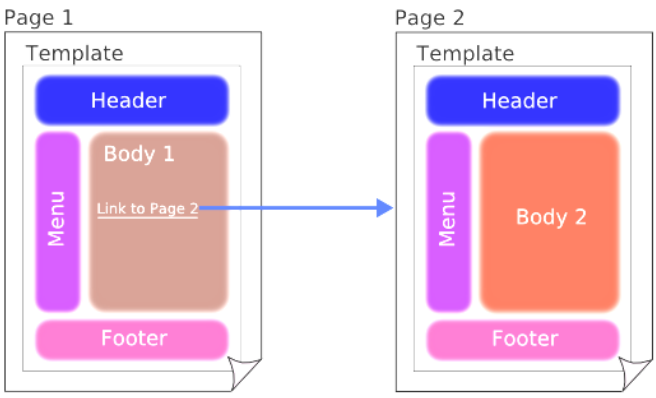


Figura 4-22. Esquema de funcionamiento de Tiles [73]

Tiles se integra fácilmente con Struts, dándolo de alta como plugin en el archivo struts-config.xml

```
        <plug-in className="org.apache.struts.tiles.TilesPlugin">
                <set-property property="definitions-config" value="/WEB-INF/tiles-defs.xml" />
        </plug-in>
```

El archivo *tiles-defs.xml* contiene una descripción de las partes que va a contener cada una de las vistas. Finalmente, cuando usamos tiles, cada JSP describe una composición de la plantilla de Tiles que se está utilizando.

El empleo de Tiles permite ahorrar tiempo en el desarrollo y modificaciones de la vista en la aplicación para conseguir un aspecto homogéneo a lo largo de la navegación web. Utilizando Tiles es posible evitar el código redundante en la creación de las vistas.

4.3.4. Hibernate Annotations 3

La aplicación web cuenta con una base de datos para almacenar los cursos creados para su posterior descarga o modificación. El motor de base de datos empleado es MySQL y para el acceso a la información desde Java se ha empleado el *framework* Hibernate.

Hibernate [78] es un *framework* de mapeo objeto relacional (ORM). Consiste en un conjunto de librerías que nos permiten acceder a los datos en las tablas como si fueran objetos. De manera que una vez creado un objeto Object o, guardarlo en la base de datos (persistirlo) es una operación tan simple como `o.persist()`.

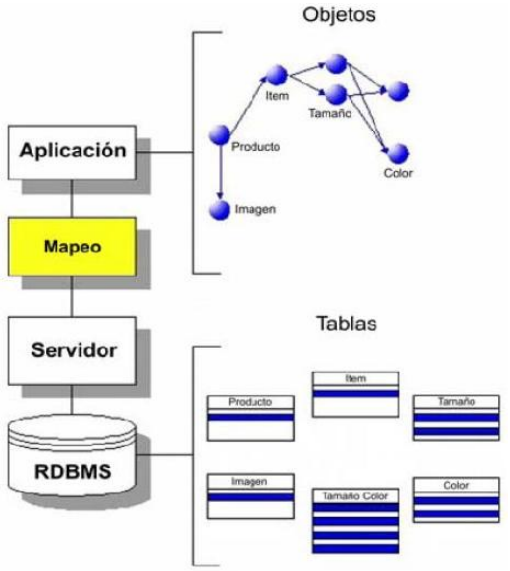


Figura 4-23. Esquema de mapeo objeto relacional ORM

En la figura anterior se aprecia que los objetos mantienen entre ellos relaciones complejas de composición y asociación, mientras que las tablas relacionales presentan un modelo matemático más estricto de relaciones entre tablas. Existen diferentes alternativas para solventar las diferencias entre el *universo de objetos* y el *universo relacional*, de hecho es posible diseñar un mapeador ORM a medida. Concretamente en la aplicación web se ha empleado Hibernate 3 en formato de anotaciones.

Realizar el mapeo de las tablas en clases anotadas de Hibernate es un proceso que se puede realizar a mano o utilizando las *Hibernate Tools.*

Estas herramientas incluyen generadores de código mediante el uso de ingeniería inversa de modo que a partir de una base de datos ya creada son capaces de obtener los archivos de mapeo. El siguiente fragmento de código muestra el código de la clase `Cursos.java` que mapea la tabla CURSOS.

```
@Entity
@Table(name = "cursos", catalog = "pld_db")
public class Curso extends BaseObject implements java.io.Serializable {

   private static final long serialVersionUID = 1L;
   private Integer id;
   private String nombreCurso;
   private String pedagogia;
   private String identifier;

   public Curso() {
           id=0;
           identifier = ""+UUID.randomUUID();
   }

   public Curso(String nombreCurso, String pedagogia, String identifier) {
           this.nombreCurso = nombreCurso;
           this.pedagogia = pedagogia;
           this.identifier = identifier;
   }

   @Id
   @GeneratedValue(strategy = IDENTITY)
   @Column(name = "id", unique = true, nullable = false)
   public Integer getId() {
           return this.id;
   }

   public void setId(Integer id) {
           this.id = id;
   }

   @Column(name = "nombreCurso", nullable = false)
   @NotNull
   public String getNombreCurso() {
           return this.nombreCurso;
   }

   public void setNombreCurso(String nombreCurso) {
           this.nombreCurso = nombreCurso;
   }

   @Column(name = "pedagogia", nullable = false)
   @NotNull
   public String getPedagogia() {
           return this.pedagogia;
   }

   public void setPedagogia(String pedagogia) {
           this.pedagogia = pedagogia;
   }

   @Column(name = "identifier", nullable = false)
   @NotNull
   public String getIdentifier() {
           return this.identifier;
   }

   public void setIdentifier(String identifier) {
           this.identifier = identifier;
   }

}
```

Para trabajar con Hibernate en el proyecto, también es necesario crear un archivo de configuración `hibernate.cfg.xml` que almacene las opciones de conexión a la base de datos y la ubicación de las diferentes clases de mapeo ORM.

```
<hibernate-configuration>
   <session-factory>
           <!-- SQL dialect -->
           <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
           <!-- Database connection settings -->
           <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
```

```
            <property name="connection.url">jdbc:mysql://localhost:3306/pld_db</property>
            <property name="connection.username">root</property>
            <property name="connection.password"></property>

            <mapping class="src.model.entity.Curso" />
            <mapping class="src.model.entity.ActividadesPedLin" />
            <mapping class="src.model.entity.BlocksPedConnectionist" />
            <mapping class="src.model.entity.MarksPedConnectionist" />
            <mapping class="src.model.entity.PhasesPedProlife" />
            <mapping class="src.model.entity.PartsPedProsim" />
            <mapping class="src.model.entity.ProblemsPedMason" />
            <mapping class="src.model.entity.ProblemsPedPbl" />
            <mapping class="src.model.entity.ProblemsPedPolya" />
            <mapping class="src.model.entity.ResourcesPedConnectionist" />
            <mapping class="src.model.entity.ResourcesPedMason" />
            <mapping class="src.model.entity.ResourcesPedPbl" />
            <mapping class="src.model.entity.ResourcesPedPolya" />
            <mapping class="src.model.entity.ResourcesPedProlife" />
            <mapping class="src.model.entity.ResourcesPedProsim" />
            <mapping class="src.model.entity.Step3PedMason" />
            <mapping class="src.model.entity.Step3PedPolya" />
            <mapping class="src.model.entity.SubpartsPedProsim" />
    </session-factory>
</hibernate-configuration>
```

4.3.5. Spring Framework

Spring es un amplio *framework* que cubre varios aspectos de un desarrollo de software [79]. Incluye incluso su propio patrón MVC a modo de Struts. En la siguiente figura se muestran los diferentes componentes que proporciona Spring.

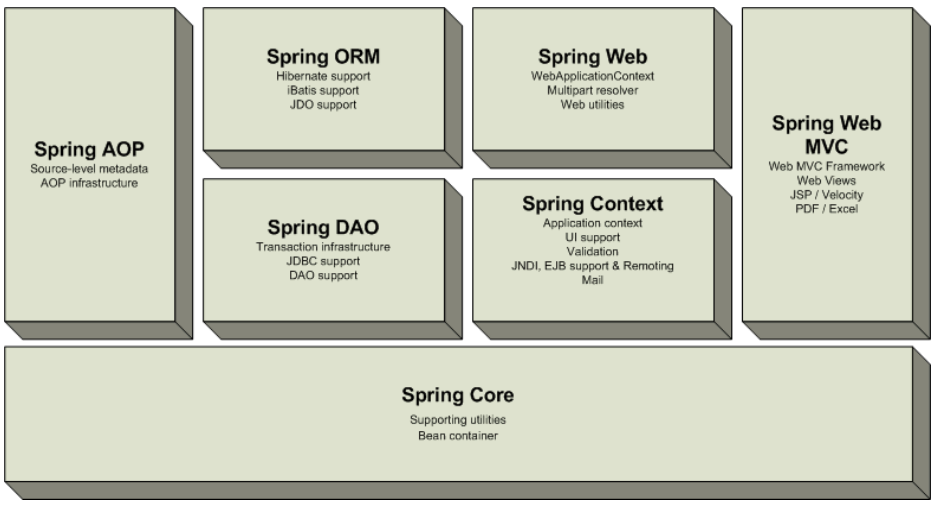


Figura 4-24. Esquema de mapeo objeto relacional ORM

En el caso de esta aplicación, presenta dos características que lo hacen realmente útil para la creación de una arquitectura de capas y la gestión de las transacciones de Hibernate:

- Inversión de Control (IoC). Spring proporciona un contexto capaz de almacenar un conjunto de beans que se inyectan durante el tiempo de ejecución. Es lo que también se conoce como inyección de dependencias. En una aplicación diseñada en varias capas (como la herramienta de

autor PeDaLea), Spring proporciona la posibilidad de independizar por completo unas capas de otras. Spring se coloca como controlador principal y gestor de los beans de la aplicación.

- Programación orientada a aspectos (AOP). La programación orientada a aspectos es un paradigma de programación cuyo objetivo fundamental es incrementar la modularidad permitiendo la separación de cross-cutting concerns. Permite en general crear servicios de manera declarativa y en lo que aplica a este proyecto, va a facilitar la gestión de las transacciones de Hibernate, que es el concern más habitual que gestiona Spring.

La integración de Struts-Spring-Hibernate puede realizarse de diferentes maneras. El primer aspecto es identificar lo que aporta al proyecto web cada *framework*.

Struts se encargará del manejo de las request de los usuarios para derivar el control a las diferentes Actions, hacer las llamadas a la lógica de negocio y preparar las diferentes vistas mediante el uso de ActionForms (ayudadores de la vista) y beans de Java en forma de Data Transfer Objects (DTO).

Hibernate será la capa de acceso a datos y permite abstraer al desarrollador del uso de SLQ y reemplazarlo por el manejo de objetos persistentes.

4.3.6. Integración Struts-Spring-Hibernate

Spring se emplea como integrador de Struts e Hibernate. Por un lado mediante la inyección de beans es posible independizar las capas de MVC, servicio y DAO. Por otro lado, la AOP permite manejar las transacciones de Hibernate de manera que libera al desarrollador de estar permanentemente preocupado por abrir y cerrar sesiones.

La gestión de las transacciones de Hibernate es un aspecto especialmente interesante por la forma en la que trabaja el cargador de objetos de Hibernate.

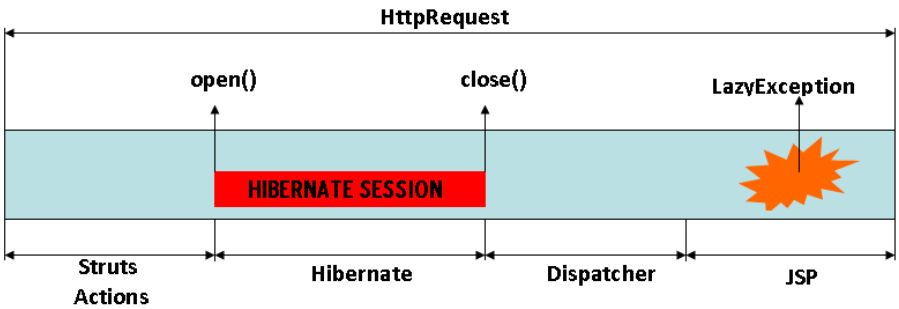


Figura 4-25. Diagrama de ocurrencia de LazyException al utilizar Hibernate

En la figura anterior se ve el proceso de una *request* a través de las diferentes capas de a aplicación. La request es gestionada inicialmente por el controlador principal (en este caso Struts) que acaba haciendo una llamada a un DAO de Hibernate que abre y cierra una sesión de Hibernate.

Durante el tiempo que la sesión permanece abierta es posible recuperar información de la base de datos mediante consultas HQL así como persistir objetos.

Imaginemos el siguiente caso. Recuperamos a través de su identificador único un objeto User que contiene una propiedad Profile. En lo que permanece abierta la sesión podríamos acceder al objeto Profile de la siguiente forma:

```
User usuario = servicioUser.getUser(id);
String language = usuario.getProfile().getLanguage();
```

La forma en la que ejecutará por defecto Hibernate las anteriores líneas es realizando dos consultas SELECT, una de ellas para recuperar la información del usuario y otra cuando se accede a la propiedad language del objeto profile. Esta forma de trabajo es lo que se llama lazy loading porque Hibernate rellena los objetos a medida que lo va necesitando dentro de la sesión.

Siguiendo con el ejemplo anterior, imaginemos que el objeto usuario traspasa varias capas de la aplicación (DTO) y llega al JSP como atributo dentro del HttpSession para mostrar la información de otra propiedad del User, por ejemplo el nombre de la empresa que está dentro del objeto Company.

En el JSP se podría escribir algo como:

```
<p>El usuario ${usuario.name} pertenece a la empresa: ${usuario.company.name}</p>
```

Lo que sucede en este caso es que estamos accediendo a una propiedad del objeto persistente cargado de forma lazy fuera de la HibernateSession, lo que provoca una excepción de tipo LazyLoadingException al intentar acceder a la información fuera de una transacción de Hibernate.

Existen múltiples alternativas para solucionar este problema. Una de ellos es utilizar un cargador eager en lugar de un cargador lazy. Para ello se usan anotaciones en los beans poniendo lazy=false en la anotación de @Proxy. De esta forma Hibernate rellena las propiedades tipo objeto de los objetos persistentes.

Otra forma de solucionar los problemas de lazy loading es especificar fetch="join" en la etiqueta de mapeo y programar las sencias HQL de esta manera:

```
from Maestro as maestro left join fetch maestro.detalle
```

Sin embargo, la solución implementada en este proyecto es la que implementa el patrón *"open session in view"* que consiste en mantener abierta la sesión de Hibernate durante el tiempo que la request se está procesando a lo largo del servidor. Concretamente, se emplean demarcaciones transaccionales utilizando AOP de Spring para mantener una única sesión abierta para todas las transacciones hasta que la vista es renderizada para entregarla al cliente.

Spring se puede integrar dentro de una aplicación web MVC Struts de diferentes maneras, la forma utilizada en este proyecto es a través de un *plugin* de Struts, declarado en el archivo de configuración de la aplicación web (*web.xml*).

```
<context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath*:applicationContext-*.xml</param-value>
</context-param>
<listener>
        <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

4.3.7. Herramientas para desarrollo y pruebas

A continuación se citan las herramientas que han sido utilizadas durante el desarrollo y pruebas de la Herramienta de Autor.

Eclipse Ganymede <http://www.eclipse.org>

Eclipse es el IDE más conocido para desarrollo en Java. Es un software de código abierto y existe una multitud de contribuciones que extienden su funcionalidad para adaptarse al desarrollo con los diferentes *frameworks* de Java (*Struts, Hibernate, Spring*…). Una de las extensiones más conocidas y utilizada en este proyecto son las *JBoss Hibernate Tools* incluidas dentro del paquete de *Seam Tools*.

Exadel Studio Pro <http://www.exadel.com/web/portal/products/ExadelStudioPro>

Exadel™ Studio Pro es un plugin de que permite convertir a Eclipse en un entorno completo para el desarrollo de aplicaciones web. Exadel incorpora herramientas (como un servidor Tomcat integrado para pruebas) que facilitan el desarrollo de aplicaciones web con Struts, J2EE y AJAX dentro del entorno de Eclipse.

MySQL Tools <http://www.mysql.com/products/tools>

Son un conjunto de herramientas que facilitan la tarea creación de tablas en la base de datos. También facilitan el trabajo a la hora de realizar consultas. Son las siguientes: MySQL Migration Toolkit, Administrator, Query Browser, Workbench.

Visual Paradigm for UML <http://www.visual-paradigm.com>

Es una *suite* de productos para entornos de desarrollo de software con el que es posible realizar entre otras cosas, diagramas de casos de uso, esquemas de base de datos, diagramas de clases… Requiere licencia y para este proyecto se ha utilizado un *trial key* que se provisiona para usuarios registrados.

CopperCore v3.1 <http://coppercore.sourceforge.net>

Es el primer *player* de IMS LD para los niveles A, B y C. Su utilización durante el desarrollo del proyecto ha sido muy útil a la hora de verificar la validez de las UoL generadas mediante la herramienta de autor.

Reload LD Player <http://www.reload.ac.uk/ldplayer.html>

Se trata de otro *player* de *Learning Design*. Está basado en las especificaciones de IMS LD y hace uso del motor de Coppercore. La utilización de Reload LD Player permite ejecutar un *play*. El usuario elige el rol que va a desempeñar y se sucede la secuencia del curso. Reload LD Player mejora a CopperCore v3.1 en el aspecto de creación y gestión de usuarios y roles.

VMWARE y GRAIL

La Universidad Carlos III proporciona una máquina virtual de incorpora el .LRN con GRAIL. La máquina virtual se puede reproducir la herramienta gratuita VMPlayer.

5. PRUEBAS REALIZADAS EN LA APLICACIÓN

Durante el desarrollo de la aplicación se han probado los archivos generados para todas las pedagogías por la aplicación tanto en el CopperCore como en GRAIL. Las pruebas de validación XML han resultado satisfactorias pero entre ambos RTEs se han detectado ciertas diferencias de interpretación. Algunos comportamientos de los RTEs no han resultado ser los esperados para pedagogías complejas o con implementación de bucles.

En los siguientes apartados se expone un breve resumen de las pruebas de instanciación de UoLs realizadas durante el desarrollo del proyecto.

5.1. EJECUCIÓN DE UOL CON COPPERCORE

CopperCore es un *runtime* de J2EE útil para realizar pruebas pero quizá no lo suficientemente maduro como para entornos de producción. Aunque soporta la especificación IMS-LD hasta el nivel C, no hay implementación para algunos de los aspectos más relevantes de algunas de las pedagogías desarrolladas como por ejemplo los foros de discusión. Por otro lado, la gestión de los usuarios en CopperCore se realiza a través de una interfaz de comandos que si bien no es demasiado complicada de utilizar, sí que resulta ineficiente para la gestión de grandes grupos de usuarios.

Una vez desplegado en local, el acceso para la interfaz de CopperCore es la siguiente URL:

http://localhost:8080/Publisher/publication.html

Figura 5-1. Pantalla principal para carga de IMD LD en CopperCore

Al hacer click en la opción de “Publish” comienza la validación del LD. Si la validación resulta satisfactoria aparecerá una pantalla como la siguiente:

Time	Level	Message
0	INFO	Validation started.
0	INFO	step 1 - Analysing package (C:\Users\fjmurcia\Desktop\ccrt\data\upload\Redes y sistemas.zip).
7	INFO	step 2 - validating the manifest.
86	INFO	step 3 - validating global content.
92	INFO	step 4 - checking if all files in package are referenced.
92	INFO	Validation passed successfully.
92	INFO	Start processing manifest
95	INFO	Processing manifest started
280	INFO	Successfully build component model
280	INFO	Semantic validation was succcessful
447	INFO	Processing manifest succeeded
450	INFO	Storing local webresources in C:\Users\fjmurcia\Desktop\ccrt\\jboss-4.0.4.GA\server\default\deploy\jbossweb-tomcat55.sar\ROOT.war\0\
462	INFO	Resources are stored.

Figura 5-2. Pantalla de resultado de la validación

Mediante el uso de la interfaz Clicc de CopperCore, se crean los alumnos y tutores así como la instancia del curso

```
Clicc:/>createuser alumno1
User alumno1 created
Clicc:/>dir
Uol[id=0,title="Redes y sistemas",uri="http://www.pld-project.net/uri/LD-
2f419e4181fe",contentUri="http://localhost:8080/0/"]
Clicc:/>cd 0
Clicc:/uol=0>createrun Curso2011
0
Clicc:/uol=0>cd 0
Clicc:/uol=0/run=0>addusertorun alumno1
User alumno1 added to run 0
Clicc:/uol=0/run=0>listroles
<roles identifier="0" org-identifier="6ea23d81-5e47-11e0-9b40-ee8bfaed78d8">
  <learner identifier="1" org-identifier="Learner">
    <title>Learner</title>
  </learner>
  <staff identifier="2" org-identifier="Staff">
    <title>Teacher</title>
  </staff>
</roles>

Clicc:/uol=0/run=0>cd alumno1
Clicc:/uol=0/run=0/user=alumno1>addusertorole 1
User alumno1 added to role 1
Clicc:/uol=0/run=0/user=alumno1>setactiverole 1
Active role for user alumno1 in run 0 set to role 1
```

Figura 5-3. Código Clicc necesario para instanciar el curso

Una vez creado el curso y asignados sus alumnos y profesores mediante Cclic, es posible comenzar la ejecución del *play* principal. Para ello habrá que navegar a la siguiente URL en local: http://localhost:8080/WebPlayer/runswitch.html

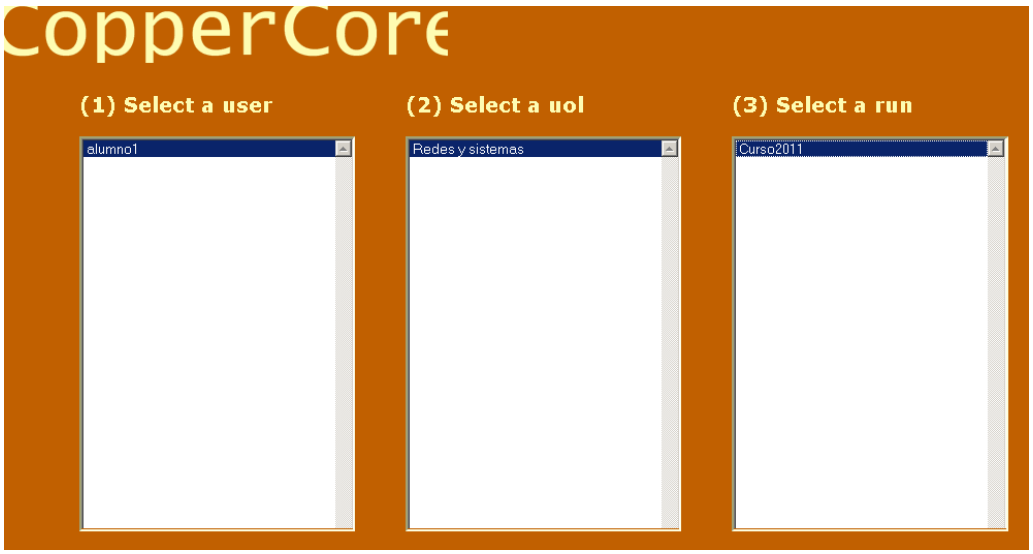


Figura 5-4. Pantalla de selección de Play de CopperCore

Como el usuario alumno1 tiene el rol de *Learner* y *Staff*, puede simular el curso completo. Para el curso de este ejemplo de pedagogía basada en problemas, el alumno visualiza primero el enunciado del problema para posteriormente participar foros de debate y subir una solución al problema.

Figura 5-5. Vista de lectura del enunciado para los alumnos en PBL

Figura 5-6. Vista de acceso a la participación en foros en PBL

En esta pedagogía, los alumnos participan en la creación de un enunciado conjunto que resume lo que han ido extrayendo de las actividades anteriores. Este enunciado es validado por parte del profesor que es el que determina un enunciado final y el pase a la segunda fase del curso. En la siguiente figura se muestra la *support activity* a través de la cual el profesor elige un enunciado final:

Figura 5-7. Upload por parte del alumno de la solución

Los alumnos visualizan el enunciado definitivo del problema y acceden a la segunda fase del curso, en el que volverán a participar en foros, y finalmente realizarán un *upload* con la solución al problema.

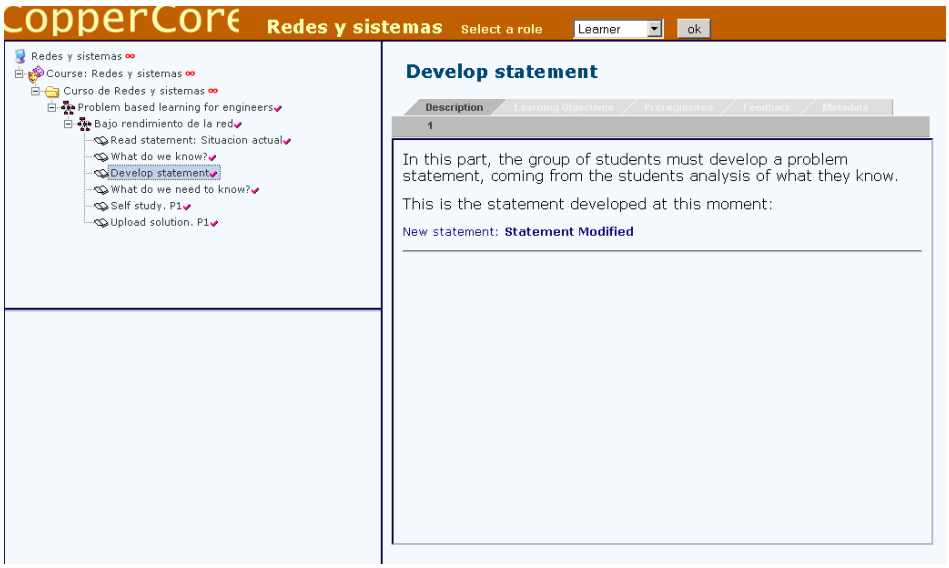


Figura 5-8. Upload por parte del alumno de la solución

En la siguiente pantalla se puede comprobar cómo el alumno para la pedagogía PBL sube a la plataforma los comentarios y el documento de la solución del problema enunciado.

Figura 5-9. Upload por parte del alumno de la solución

Finalmente, el profesor monitorizará el resultado de estos *uploads* a través de una actividad de tipo monitor:

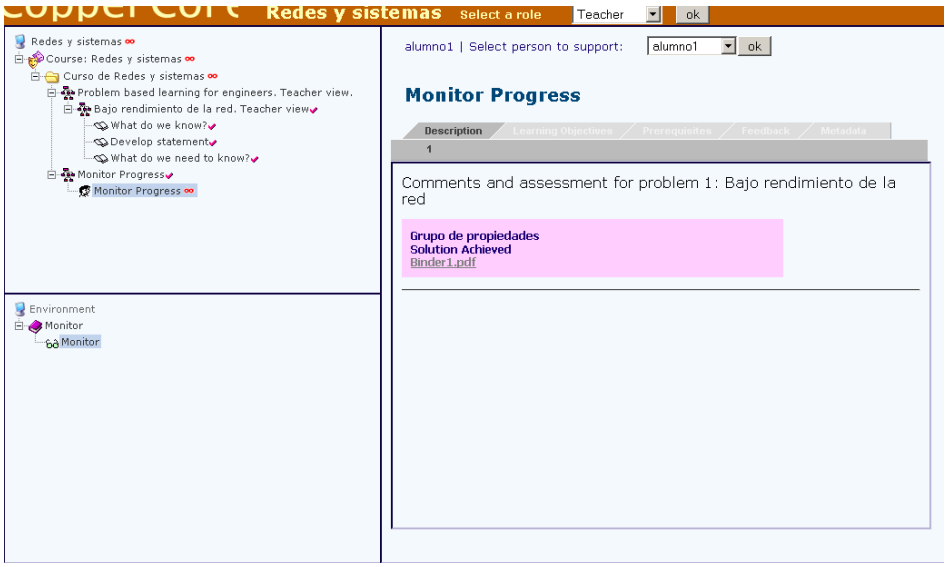


Figura 5-10. Actividad Monitor desde la que el profesor revisa los *upload* de los alumnos

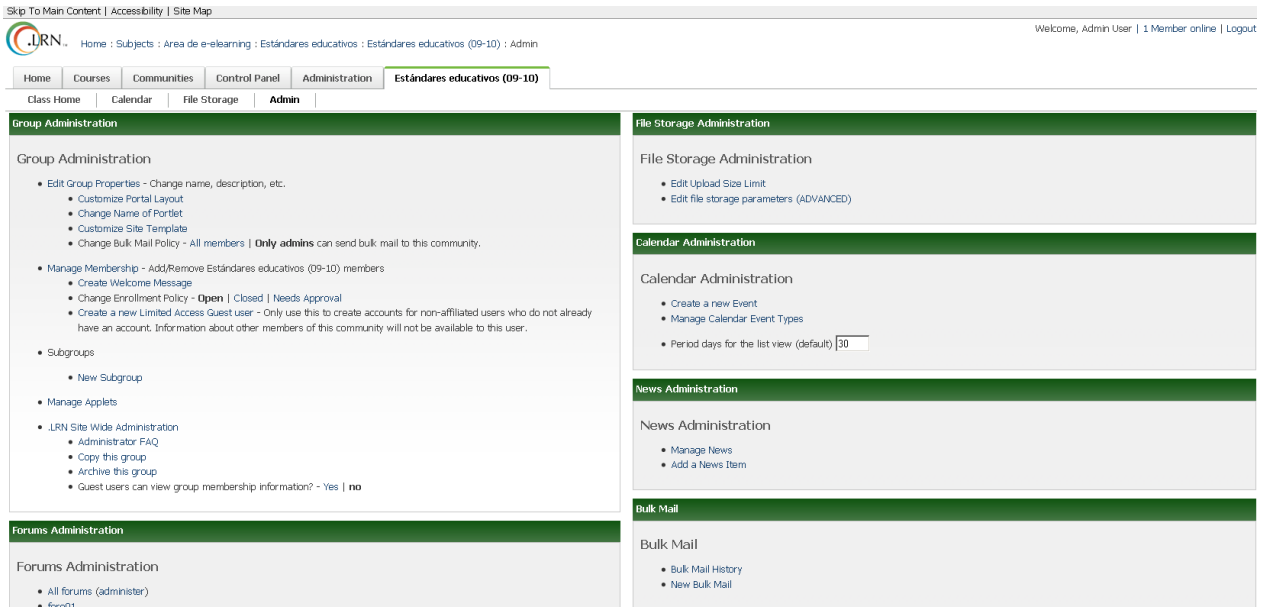
Por su parte, GRAIL al estar integrado en el LMS .LRN, proporciona acceso a una mayor cantidad de servicios y una mayor sencillez a la hora de crear y gestionar usuarios. Sin embargo, GRAIL se ha mostrado inestable en algunas de las pedagogías que utilizaban propiedades y foros de discusión.

Los detalles sobre implementación de foros con IMS-LD para GRAIL se muestran en el apartado 5.3.

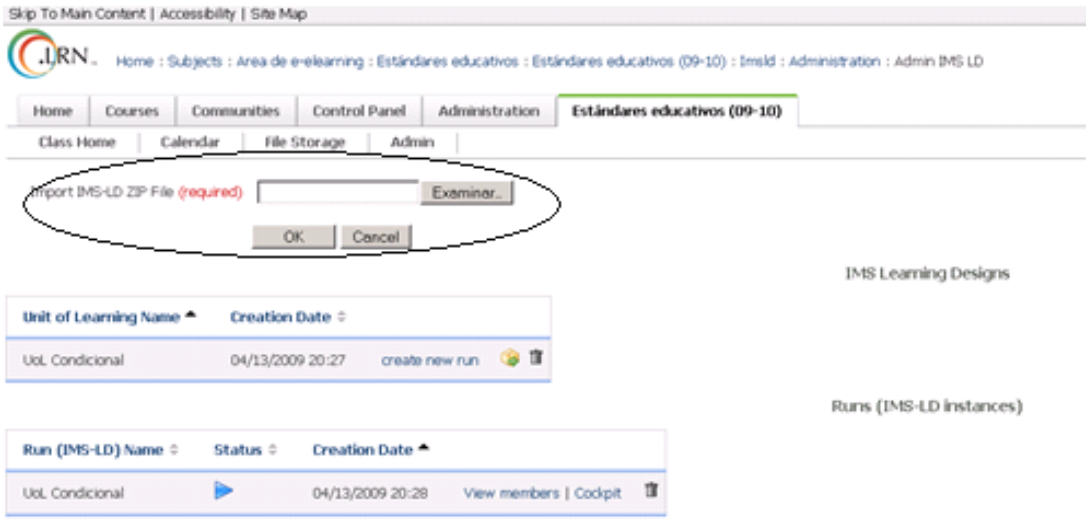
5.2. EJECUCIÓN DE UOL USANDO GRAIL

En este apartado se muestra la ejecución de una unidad de aprendizaje de la pedagogía orientada a problemas (apartado 3.5, página 43)

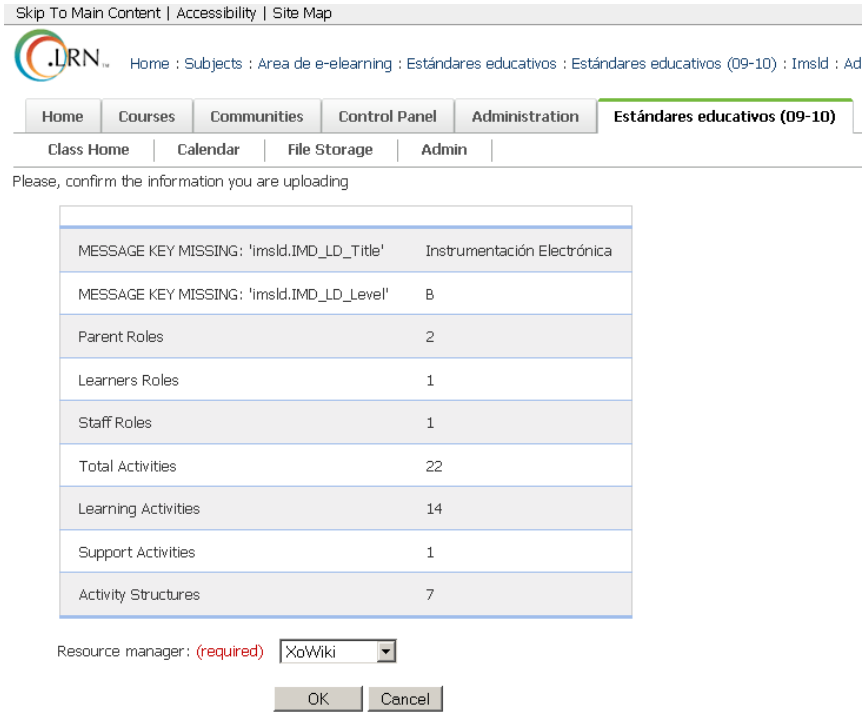
En esta demostración, se crea un nuevo curso de *Instrumentación Electrónica* a partir del archivo ".zip" que contiene el *manifest* de IMS-LD y que ha devuelto la plataforma. En la siguiente figura se muestra la pantalla de configuración de una comunidad en .LRN.



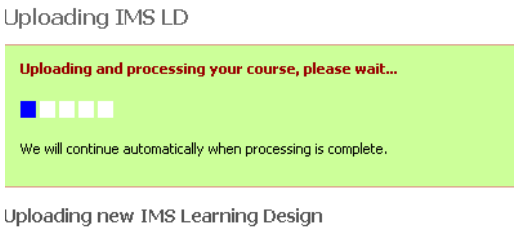
Dentro del curso ir a "*Admin*" y en la parte de abajo del todo hay una opción *UoL administration*. Aparece el siguiente menú:



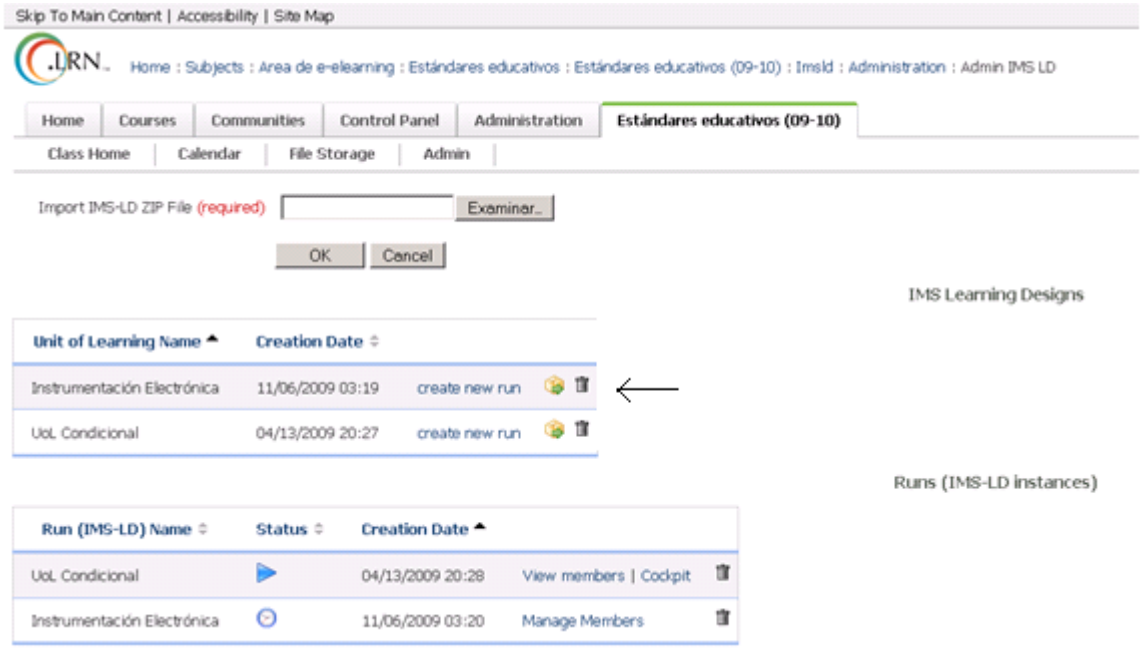
Cargar el ZIP con el nuevo curso. Si todo está correcto nos muestra la información de los elementos que contiene el curso:



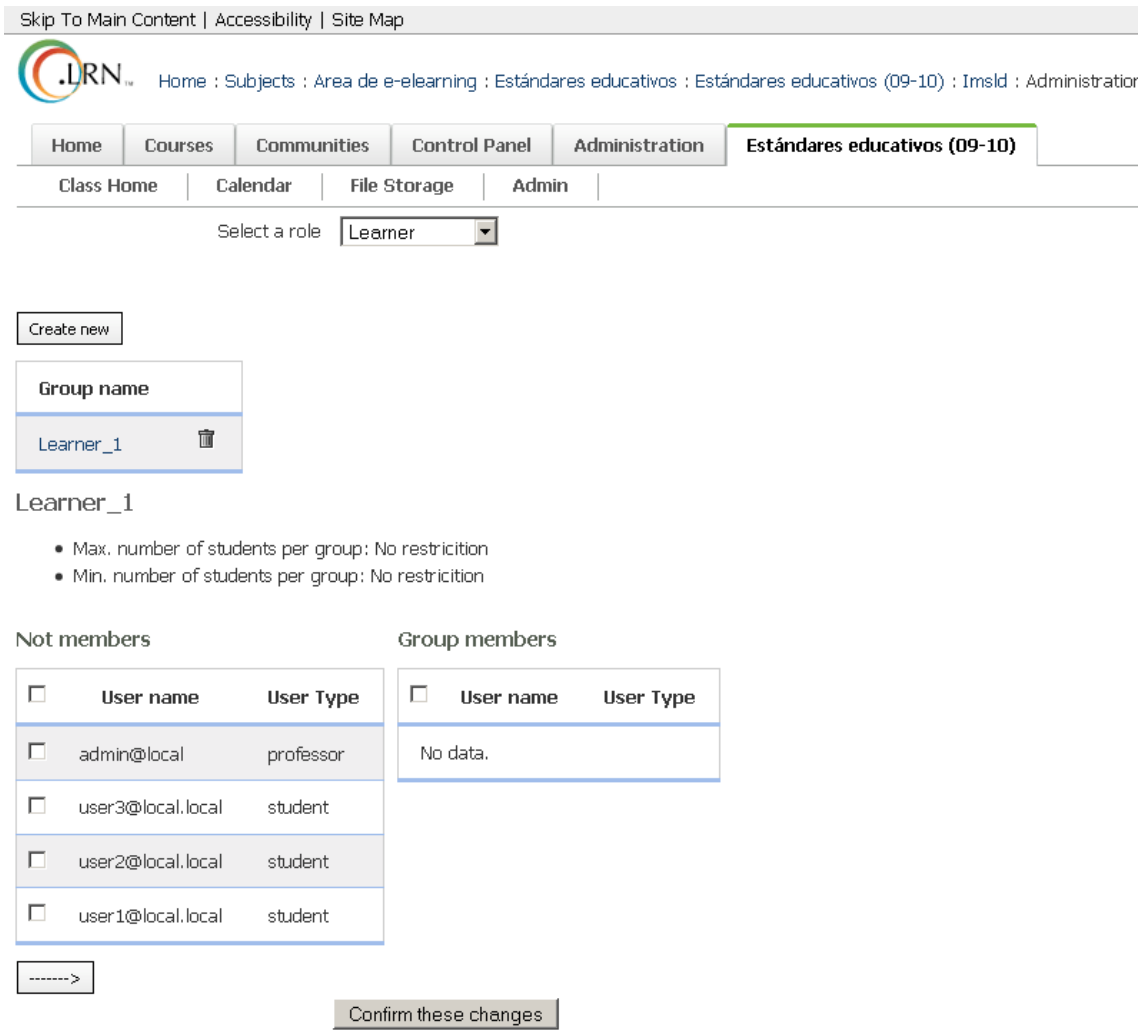
Al darle al OK aparece la validación. Los cursos generados con la herramienta PeDaLea también han sido comprobados con la utilidad de validación desarrollada por la Universidad Pompeu Fabra *IMS Doc Validator* [73].



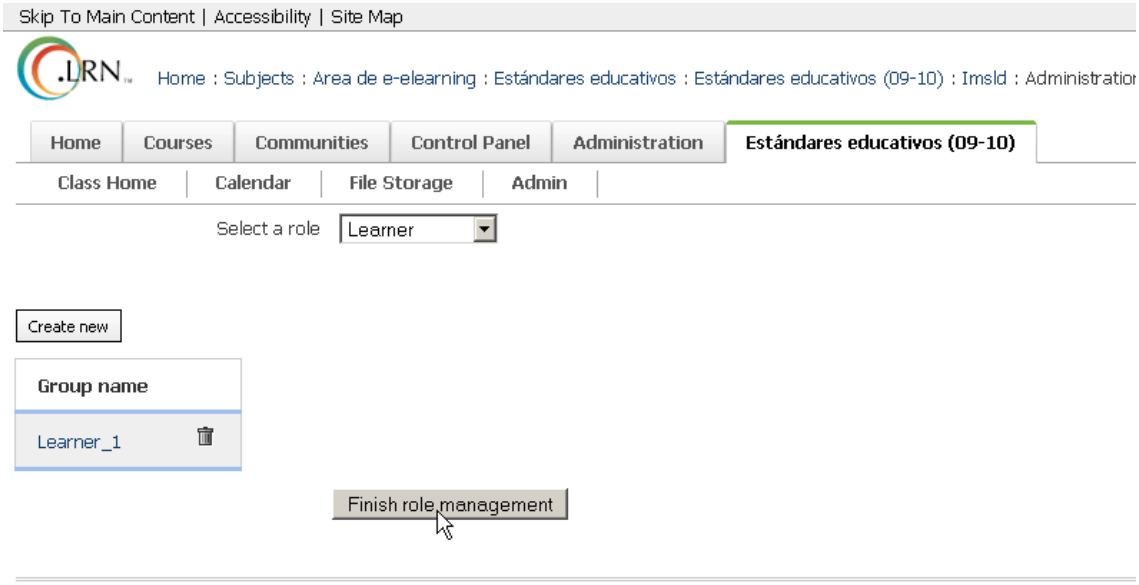
Al haberse procesado correctamente el *manifest* de IMS-LD aparece la nueva UoL subida y una *run* que se ha creado automáticamente.



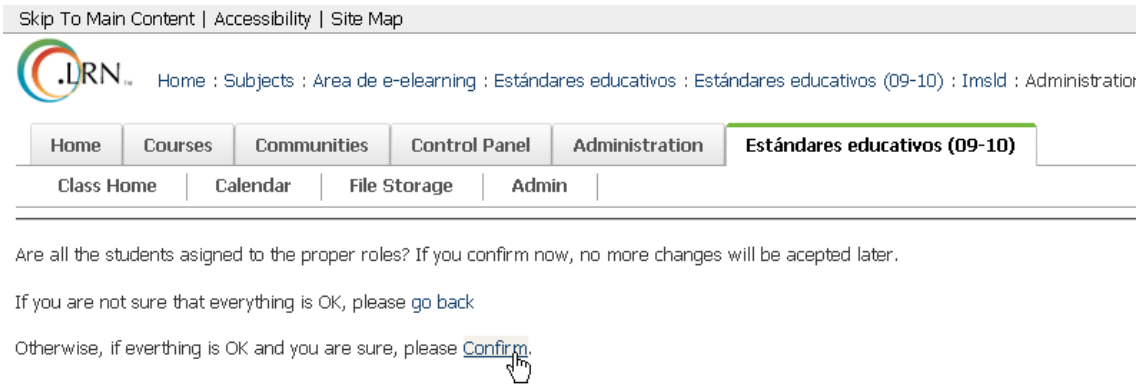
El siguiente paso es "*Manage Members*". Aparecemos en unos formularios que nos proporcionan la facilidad de añadir nuevos usuarios en sus roles respectivos:



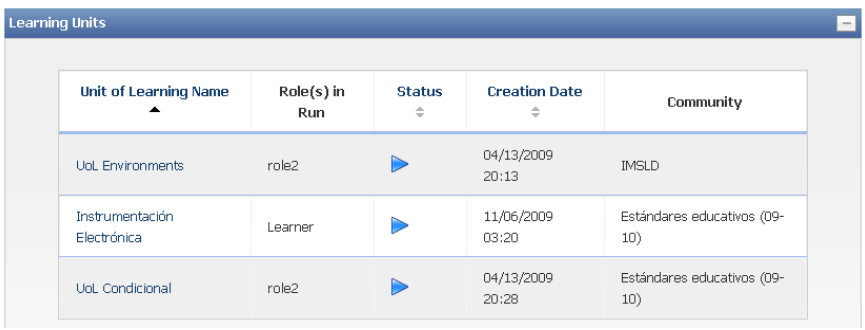
Una vez terminado de añadir cada usuario a cada rol, podemos darle a “Finish”



La plataforma nos pregunta si estamos conformes con los usuarios y roles que hemos creado. Si todo está correcto damos a confirmar:

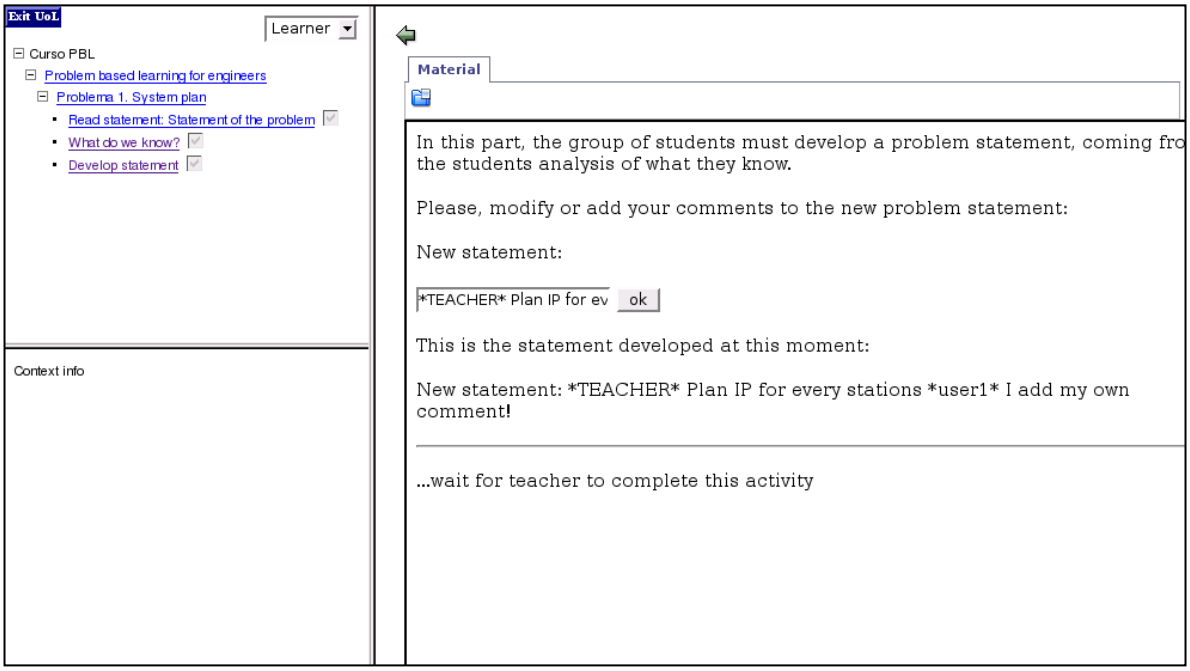


Cuando uno de los usuarios accede a la plataforma ya podría ver el run del curso disponible para empezar:

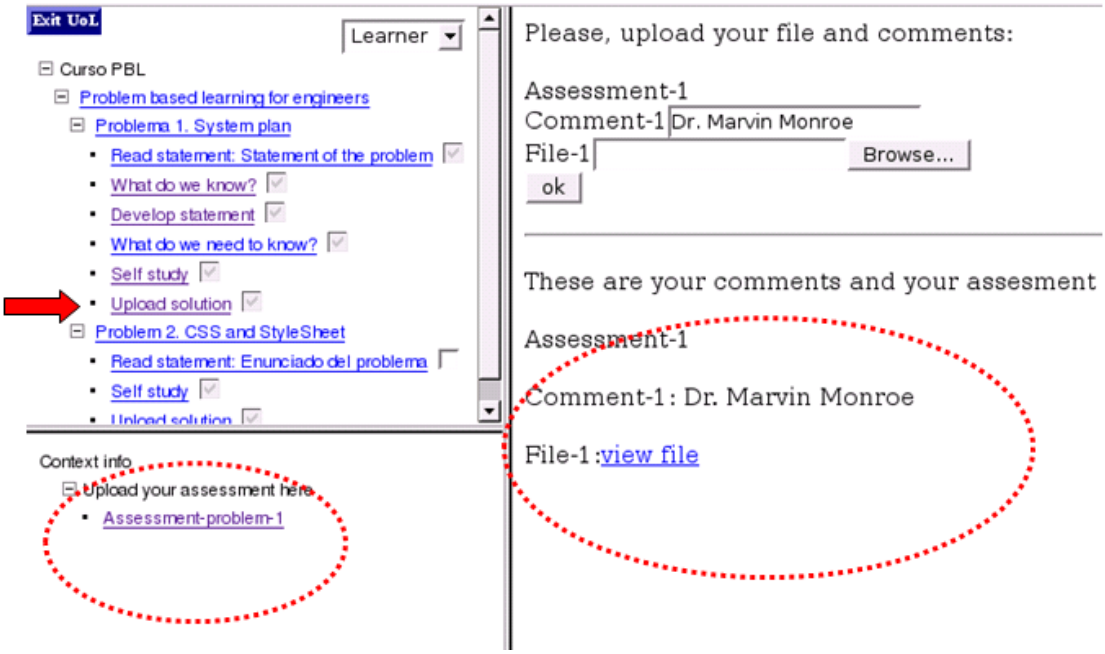


Las siguientes capturas de pantalla pertenecen a la ejecución de la UoL desplegada: *Curso PBL.*

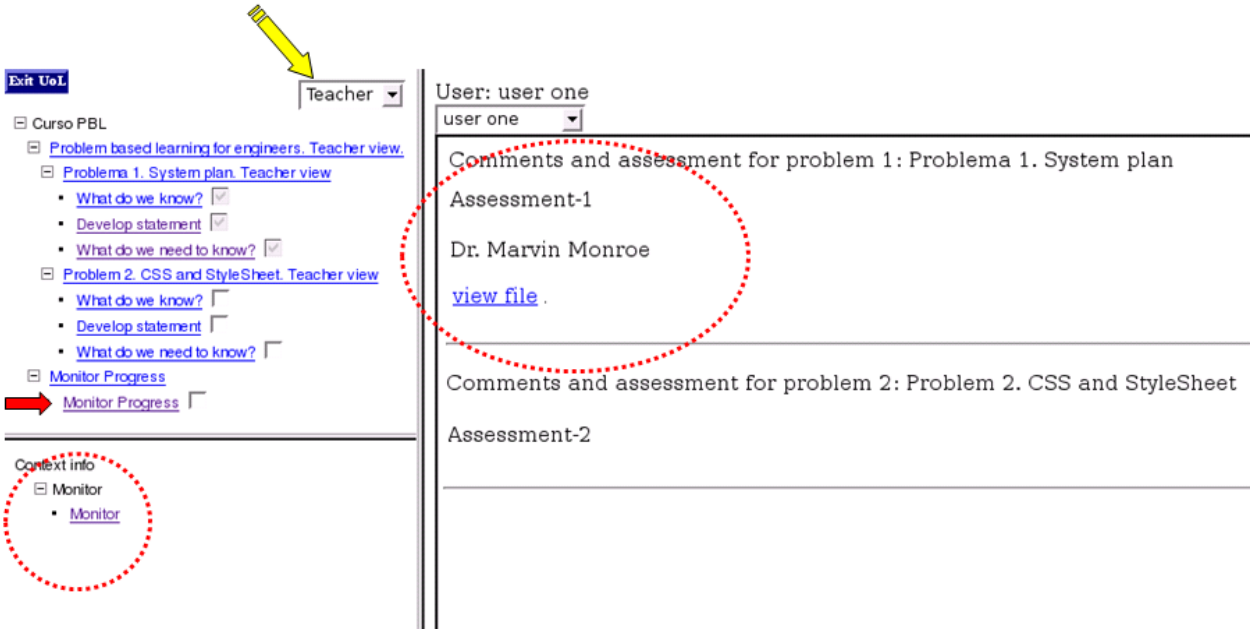
Al acceder al curso como *learner* se van mostrando las actividades que los alumnos deben ir realizando. Al llegar a un punto en concreto del curso el progreso del alumno se detiene hasta que el profesor de por finalizada esa actividad y permita a los alumnos continuar. En el pantallazo mostrado abajo se muestra la vista del alumno:



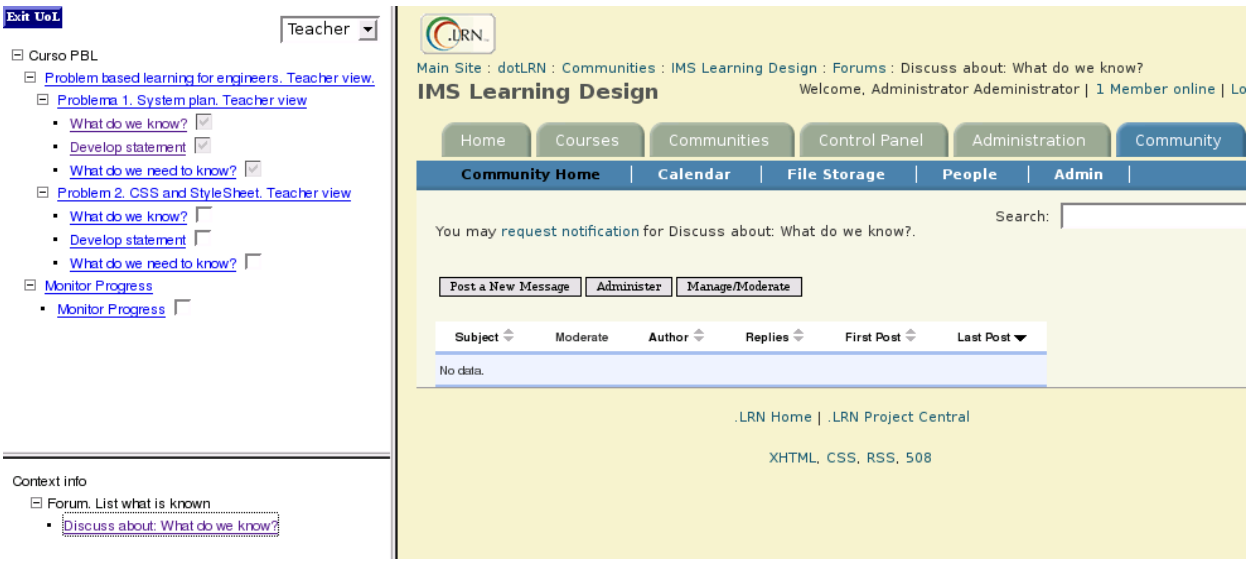
Llegado el momento, el alumno sube la solución del problema a través de una actividad *"Assesment-problem-1"* colocada en el *environment.* En la figura de abajo se muestra la subida del fichero y comentarios realizado por un alumno.



Al acceder al curso como *teacher* se muestran en el panel de la izquierda los recursos disponibles y un acceso al *monitor*, desde el cual se monitorizarán los datos subidos por parte de los alumnos, tanto comentarios como posibles ficheros. En la siguiente figura se muestra el pantallazo de la ejecución en .LRN para la monitorización por parte del profesor de los archivos y comentarios enviados por los alumnos.



Finalmente, en el pantallazo de abajo se muestra la visualización de un profesor cuando accede a uno de los diferentes foros de discusión que se generan con esta pedagogía.



En el siguiente apartado se habla precisamente de las particularidades de la instanciación de foros en GRAIL.

5.3. INSTANCIACIÓN DE FOROS EN .LRN

En .LRN hay un problema al instanciar foros. La forma correcta de instanciarlos desde IMS-LD es la siguiente:

```
<imsld:learning-activity identifier="LA-FORO" isvisible="true">
     <imsld:title>What do we know?</imsld:title>
     <imsld:environment-ref ref="E-Forum" />
     <imsld:activity-description>
            <imsld:item identifier="I-FORO" identifierref="RES-FORUM" />
     </imsld:activity-description>
     <imsld:complete-activity>
            <imsld:user-choice />
     </imsld:complete-activity>
</imsld:learning-activity>

<imsld:environments>
     <imsld:environment identifier="E-Forum">
     <imsld:title>Forum. List what is known</imsld:title>
     <imsld:service identifier="Serv-Foro">
            <imsld:conference conference-type="asynchronous">
                   <imsld:title>Discuss about: What do we know?</imsld:title>
                   <imsld:participant role-ref="Learner" />
                   <imsld:moderator role-ref="Staff" />
                   <imsld:item identifier="ID-CONF" identifierref="RES- FORUM" />
            </imsld:conference>
     </imsld:service>
     </imsld:environment>
</imsld:environments>

<resources>
     <resource identifier="RES-FORUM " type="forum" href="res- forum.html">
     <file href="res-step2-forum1.html" />
     </resource>
</resources>
```

- Se define una *learning-activity* y dentro de ella un *item* como referencia a un recurso con una breve explicación de lo que se espera en el foro y también una referencia a un *environment* que contiene el propio foro.
- Se define un *environment* en el que se define el servicio *conference* asíncrono y los roles de cada uno de los participantes. En este *enviroment* se pone una referencia a un *item* de tipo *forum*.
- Se definen dos recursos, uno de tipo *webcontent* que es el que contiene la explicación acerca del funcionamiento o normativa del foro. Y otro recurso de tipo *forum* que es el que genera realmente el foro.

En la siguiente figura se muestra la relación entre los elementos de IMS-LD para la correcta instanciación de un foro:

Figura 5-11. Instanciación de un foro de discusión utilizando IMS-LD

De esta manera, debería funcionar del siguiente modo:

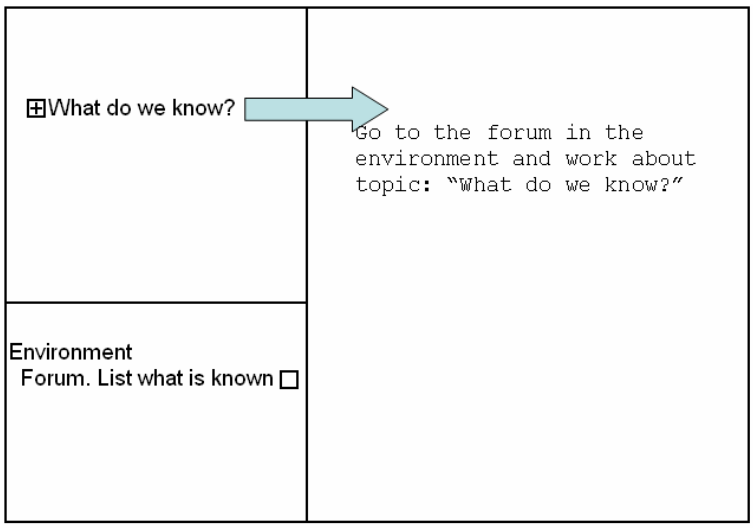


Figura 5-12. Visualización esquemática de .LRN para el recurso que muestra las instrucciones para la participación en el foro

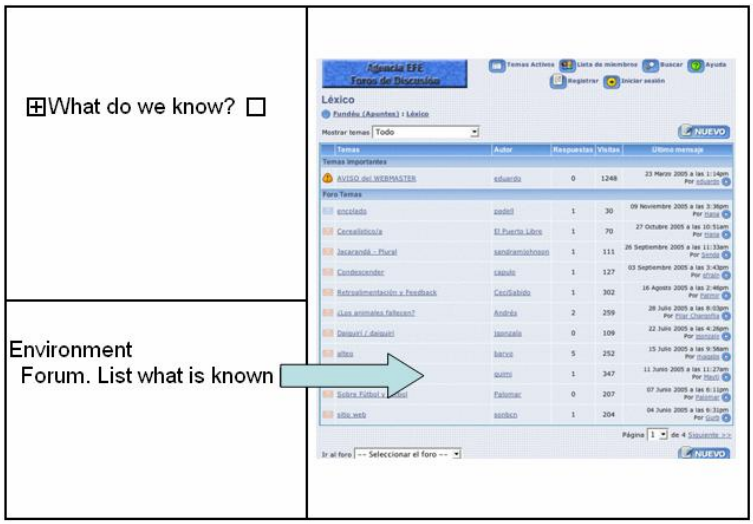


Figura 5-13. Visualización esquemática de .LRN al acceder al foro de discusión

Sin embargo, el código XML de la figura 5-11 genera un error en GRAIL y la UoL no se puede llegar a cargar. En CopperCore sí se puede cargar, al desplegarse funciona de manera similar a la mostrada en las figuras anteriores pero CopperCore no genera un foro real porque no dispone de esa funcionalidad.

Finalmente, se realizó la siguiente modificación en la instanciación de foros:

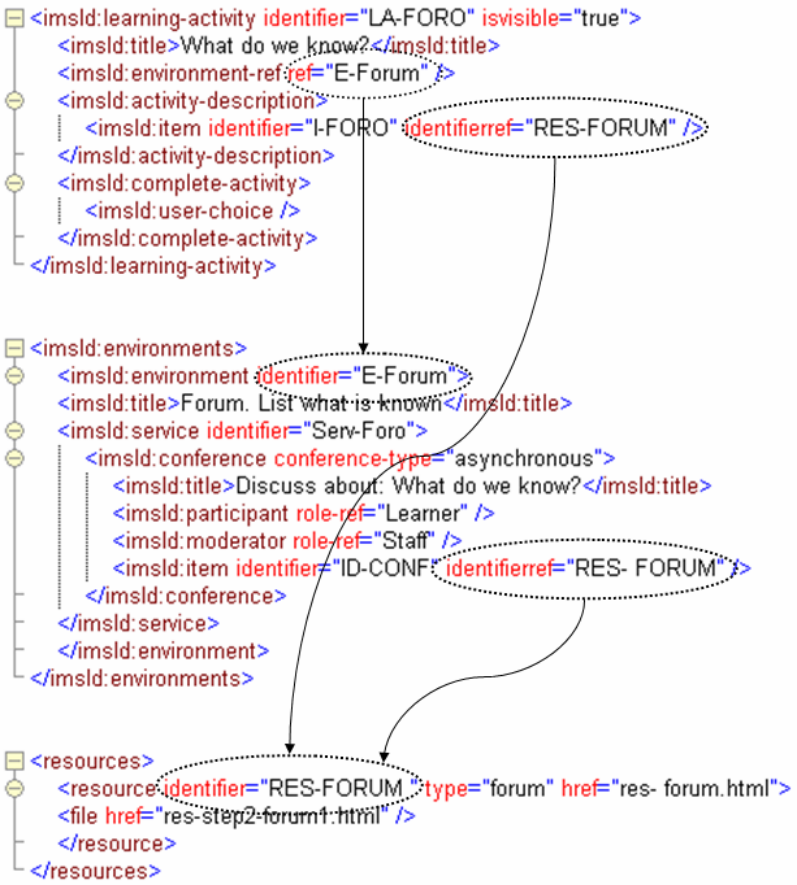


Figura 5-14. Código IMS-LD para la correcta instanciación de foros en .LRN

Como se ve, se generan dos foros, Uno en la sección de *learning-activity* y otro en la sección del *environment*. Sin embargo, es la forma en la que .LRN no lanza errores al cargar el *manifest* del curso y por eso lo dejamos así. CopperCore con este código funciona de igual modo al anterior.

6. CONCLUSIONES Y TRABAJOS FUTUROS

6.1. CONSECUCIÓN DE OBJETIVOS

El resultado obtenido de este proyecto ha sido satisfactorio ya que el objetivo principal que se perseguía se ha logrado. Ha sido posible lograr el desarrollo de una herramienta de autor: PeDaLea (Pedagogical Learning Design) que está basada únicamente en términos pedagógicos y es capaz de generar contenidos IMS-LD que se pueden cargar en plataformas compatibles.

Las pedagogías que se han mapeado a IMS-LD con éxito en este proyecto han sido las siguientes:

- Pedagogía Lineal
- Método de resolución de problemas de Polya
- Método de resolución de problemas de Mason, Burton y Stacey
- Aprendizaje basado en problemas para ingeniería
- Aprendizaje basado en proyectos con roles simultáneos
- Aprendizaje basado en proyectos en ciclo de vida.

PeDaLea presenta para cada una de estas pedagogías a través de una sencilla interfaz web en la que el profesor va eligiendo los contenidos del curso. Una vez finalizada la edición se genera para cada una de estas pedagogías contenido IMS-LD válido. Las pruebas realizadas demuestran que estos ficheros generados por PeDaLea que se puede cargar y ejecutar en las plataformas compatibles.

El proceso de mapeo de estas pedagogías ha comenzado por el análisis de la interacción entre alumnos, profesores y objetos de aprendizaje. Para este análisis se han empleado esquemas de bloques para visualizar la secuencia de actividades entre los diferentes roles así como los puntos de sincronización de los participantes, en el caso de que los hubiera. El siguiente paso es la traducción de este diagrama de bloques a terminología IMS-LD y decidir los *plays, acts, contitions* y resto de elementos IMS-LD necesarios para el mapeo.

Un ejemplo de la consecución de los objetivos del proyecto es el mapeo satisfactorio de la pedagogía orientada a problemas (PBL). Para esta pedagogía, el profesor que accede a PeDaLea simplemente tiene que subir a la plataforma un archivo con el enunciado del problema que los alumnos tienen que resolver y si lo desea, algunos recursos adicionales.

A partir de esta sencilla configuración, PeDaLea es capaz de generar una compleja unidad de aprendizaje que incluye entre otras características: foros de debate, entrega de archivos por parte de los alumnos y monitorización de los alumnos por parte del tutor. Crear este tipo de unidad de aprendizaje con las herramientas disponibles en la actualidad y sin nociones sobre la especificación IMS-LD podría llegar a resultar incluso irrealizable.

Se muestra en la siguiente figura el posicionamiento de PeDaLea frente a otras herramientas de autor que también generan IMS-LD:

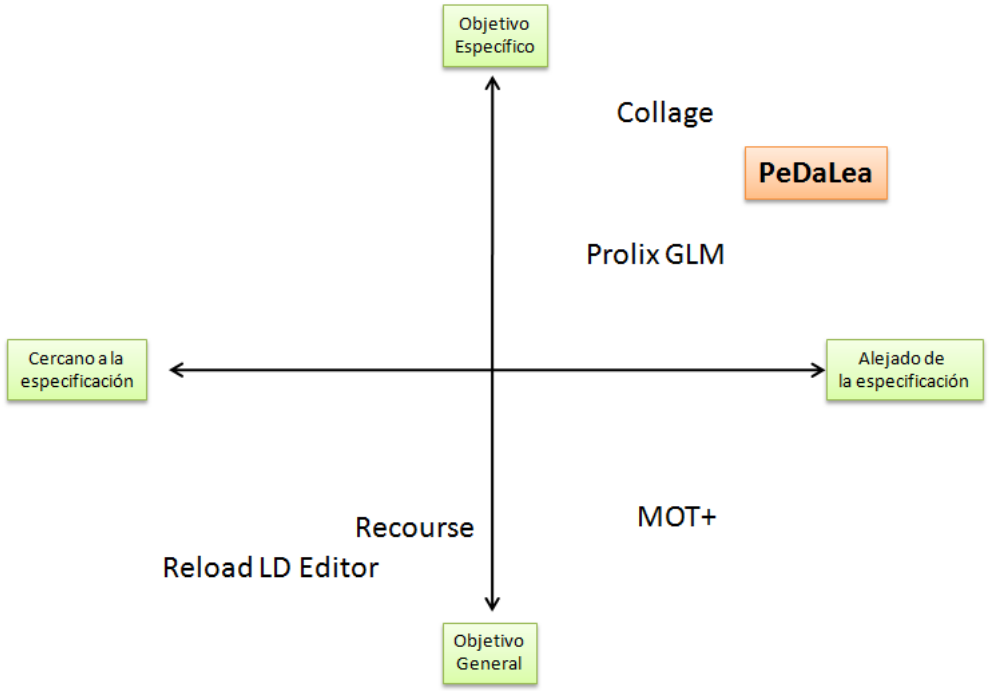


Figura 6-1. Clasificación de las herramientas de autor en función de los criterios de Oberhuemer [8]

Como se puede ver en la figura anterior, desde el punto de vista del usuario, PeDaLea está alejada de las complejidades de especificación IMS-LD y únicamente está descrita en términos pedagógicos. Este factor resulta muy positivo porque oculta a los creadores de cursos las complejidades de la especificación.

A cambio de la ocultación de la complejidad de IMS-LD, PeDaLea resulta ser una herramienta de autor de objetivo altamente específico ya que genera IMS-LD acorde únicamente las pedagogías que han sido implementadas. Para que PeDaLea genere IMS-LD con otras orientaciones pedagógicas o modifique las existentes se requiere ampliar o modificar el código fuente de la aplicación.

Con el objetivo poder simplificar las futuras ampliaciones o modificaciones, se ha optado por una arquitectura de software flexible, que se basa en los principios de inversión de control e inyección de dependencias. En el apartado 4.2.3 (pág. 60) sobre arquitectura de software se muestran los detalles sobre los aspectos técnicos PeDaLea.

6.2. CONCLUSIONES SOBRE IMS-LD

La especificación IMS-LD fue desarrollada para soportar el diseño de cursos basados en múltiples pedagogías. La flexible estructura interna de la especificación permite que esto sea posible desde un punto de vista teórico. Algunos estudios [40] [47] [52] han demostrado la capacidad expresiva de IMS-LD a la hora de adaptarse a distintos escenarios de enseñanza-aprendizaje y a lo largo de este documento se ha podido comprobar la versatilidad de la especificación para expresar diferentes situaciones educacionales. Todos estos estudios demuestran las capacidades de la especificación sobre el papel y la aplicabilidad de la metáfora del teatro [7] a muchos escenarios pedagógicos.

Sin embargo, IMS-LD ha acabado presentando importantes limitaciones a la hora de mapear ciertas pedagogías. En el apartado 2.3.2 se comentaron algunas de estas limitaciones relacionadas con la ausencia de un soporte adecuado para la implementación de bucles [50]. Esta limitación supone una restricción para expresar aquellas pedagogías que emplean la repetición de actividades para base para reforzar conocimientos. Para solventar esta limitación es posible replicar partes del código XML y durante el tiempo de ejecución guiar al alumno mediante el uso de ciertas condiciones para repetir las actividades un número determinado de veces y de cualquier manera, no es posible implementar bucles potencialmente infinitos. Esta limitación ha afectado, por ejemplo, al mapeo de la pedagogía conexionista de Edward Thorndike. Las pedagogías conexionistas y conductistas se basan que se basan en el patrón de ensayo y error lo que supone una repetibilidad de actividades. Las carencias de IMS-LD al respecto de la implementación de bucles hacen que estas pedagogías no puedan mapearse directamente.

La especificación IMS-LD se muestra demasiado restrictiva a la hora de considerar las actividades finalizadas como completadas para el resto de la ejecución del curso [7], ya que una vez finalizadas no pueden ser requeridas nuevamente a modo de refuerzo pedagógico. Con un modelo más completo que permitiera reflejar todos los posibles estados de las actividades se simplificaría el mapeo de las pedagogías que requieren personalización y adaptación durante el tiempo de ejecución [85].

Por otro lado, existe otra limitación inherente a la metáfora del teatro de IMS-LD ya que para que un alumno acceda a un nuevo bloque formativo debe finalizar el anterior, lo que no está en consonancia por ejemplo con las pedagogías basadas en aprendizaje por descubrimiento.

Durante el desarrollo de este proyecto, se ha tratado de bordear esta limitación de la especificación mediante la repetición de actividades en el código IMS-LD y el uso de condiciones para guiar al alumno en su experimento de ensayo y error pero el resultado no ha sido del todo satisfactorio. Pese a que el contenido IMS-LD pudiera ser válido desde el punto de vista del esquema XML, en tiempo de ejecución CopperCore y GRAIL han mostrado comportamientos dispares y limitados para este tipo de pedagogías.

Se han detectado ciertas discrepancias en tiempo de ejecución entre CopperCore y GRAIL para una misma UoL generada por PeDaLea. La discrepancia entre los LMS se debe en parte a que IMS-LD no cubre aspectos concretos relativos a la ejecución de una UOL de modo que el comportamiento de ambas plataformas se muestra distinto a la hora de conceder precedencia a los diferentes criterios de ocultación o muestra de actividades. De este modo, una misma UoL presenta comportamientos diferentes en CopperCore y GRAIL. Resultaría necesario fijar los detalles de interpretación y de implementación de los RTEs para tener un elemento sólido sobre el que poder desplegar las UOLs.

6.3. TRABAJOS FUTUROS Y CUESTIONES ABIERTAS

PeDalea puede ampliarse con el desarrollo de nuevas pedagogías. La capa de API de IMS-LD desarrollada puede ser utilizada para la creación de nuevos mapeos. En el Manual del Desarrollador (Apéndice C) se encuentran las explicaciones para añadir nuevas pedagogías a la aplicación web.

Una posible pedagogía a implementar podría estar basada en el refuerzo de conceptos por parte del profesor durante el tiempo de ejecución. Se trataría de un tipo de aprendizaje significativo. Para esta pedagogía durante el tiempo de diseño, el profesor elegiría los conceptos que se impartirían en el curso y sus actividades asociadas. Durante el tiempo de ejecución y en función de ciertas propiedades, los alumnos irían proporcionando información de los conceptos de los que carecen y el profesor desde su interfaz de usuario podría monitorizar este progreso y actuar en consecuencia para reforzar las carencias de cada alumno.

Otro aspecto en el que PeDaLea podría ampliarse sería en la creación de un perfilado de usuarios de manera que se incluyeran en la aplicación funciones de autenticación y autorización. De esta forma, por ejemplo, podrían existir unos usuarios con permisos para editar los contenidos de los cursos y otros con permisos únicamente para descargar los cursos creados.

Pese a que el objetivo inicial de este proyecto no era analizar la especificación IMS-LD, se han encontrado algunas posibilidades de mejora en la especificación de cara a mejorar en la expresividad pedagógica de la misma. El grupo de trabajo Grapple ha publicado en enero de 2011 una serie de propuestas de mejora a la especificación [85] para solucionar las restricciones en cuanto a la expresividad pedagógica, aprendizaje adaptativo e interoperabilidad con otras especificaciones.

Por otro lado, los RTEs deben mejorar en la integración de servicios [71] así como en incorporar elementos portables que amplíen los servicios disponibles para alumnos y profesores [86] , una implementación de este tipo de componentes a modo de *widget* son los Wookies [87]. Otro aspecto que se ha mostrado a lo largo del proyecto como posible mejora en las RTEs es la posibilidad de modificar en tiempo de ejecución los parámetros de las UoL a través de una sencilla interfaz para incrementar la flexibilidad del proceso de enseñanza *online* y conseguir mejorar la adaptabilidad de los cursos. Algunas publicaciones [88] apuntan igualmente hacia este hecho.

A lo largo de este proyecto se ha abordado la aplicación de IMS-LD al sector académico pero también es importante destacar que el sector corporativo se ha interesado por los beneficios que ofrece el diseño de *workflows* con IMS-LD. Para un gestor de proyectos puede ser interesante utilizar IMS-LD nivel C para distribuir tareas entre los integrantes del mismo. Al fin y al cabo existe un mimetismo entre el entorno docente y el entorno laboral, ya que en ambos se trata de que cada persona asuma un rol y se le presenten las actividades que debe realizar en el orden adecuado. Las posibilidades de esta aplicación de IMS-LD a la empresa pueden ser interesantes ya que sería posible aumentar la productividad de aquellas empresas que se decidieran a implantarlo.

7. PRESUPUESTO

Este presupuesto recoge las tareas, recursos y costes directos e indirectos relacionados con la implementación de una herramienta de autor generadora de IMS-LD para diferentes pedagogías.

7.1. RESUMEN DE RECURSOS Y ROLES

Para el desarrollo del proyecto se tendrán en cuenta los siguientes perfiles:

- **1 Ingeniero Senior (Tutor).** Ingeniero experto en IMS-LD y modelado de pedagogías. Se encargará de participar en el mapeo de las pedagogías a IMS-LD y actuará como consultor funcional. También se responsabilizará de la supervisión del proyecto y de la revisión de la documentación asociada.
- **1 Analista Programador (Alumno).** Se trata de un perfil técnico con conocimientos Java. Recopilará información sobre pedagogías, IMS-LD y el estado del arte. Realizará el mapeo de las pedagogías a IMS-LD y se encargará de elegir la arquitectura y los framewoks a utilizar. Desarrollará por completo la aplicación: controlador web, componentes de la vista y lógica de negocio. También será el encargado de realizar la batería de pruebas pertinente a la aplicación.

7.2. PLANIFICACIÓN DEL PROYECTO

El proyecto consiste en el desarrollo de una herramienta de software capaz de generar ficheros compatibles con la especificación IMS-LD. Para la ejecución del proyecto se han tenido en cuenta las siguientes fases:

- **Adquisición de conocimiento**. En esta fase se recopilará y asimilará aquella información relativa a las pedagogías que se van a mapear en la herramienta, la información sobre la especificación IMS-LD y las referencias tecnológicas a utilizar que en una primera fase serán Servlets, JSP y Struts para posteriormente incluir los *frameworks* Spring e Hibernate que proporcionarán a la herramienta flexibilidad y posibilidades futuras de ampliación.
- **Análisis**. Durante esta fase se realizará el análisis funcional y de requisitos de la aplicación a desarrollar así como el diseño del mapeo de las pedagogías a términos IMS-LD
- **Implementación y codificación**. En esta fase se realizará la implementación de la herramienta. Por un lado se desarrollará el esquema de base de datos necesario para el funcionamiento de la herramienta así como el controlador de la aplicación web y la lógica de negocio necesaria para la creación de los archivos XML compatibles con IMS-LD
- **Pruebas**. Se dividirán en una batería de pruebas de la propia aplicación web (validación de formularios, comprobación de errores en el controlador web…) y otra batería de pruebas para realizar el test de los archivos IMS-LD generados por la aplicación en los RTEs requeridos.
- **Documentación.** Creación de la documentación del proyecto.

En la siguiente figura se muestra la duración y escalonado de las tareas del proyecto.

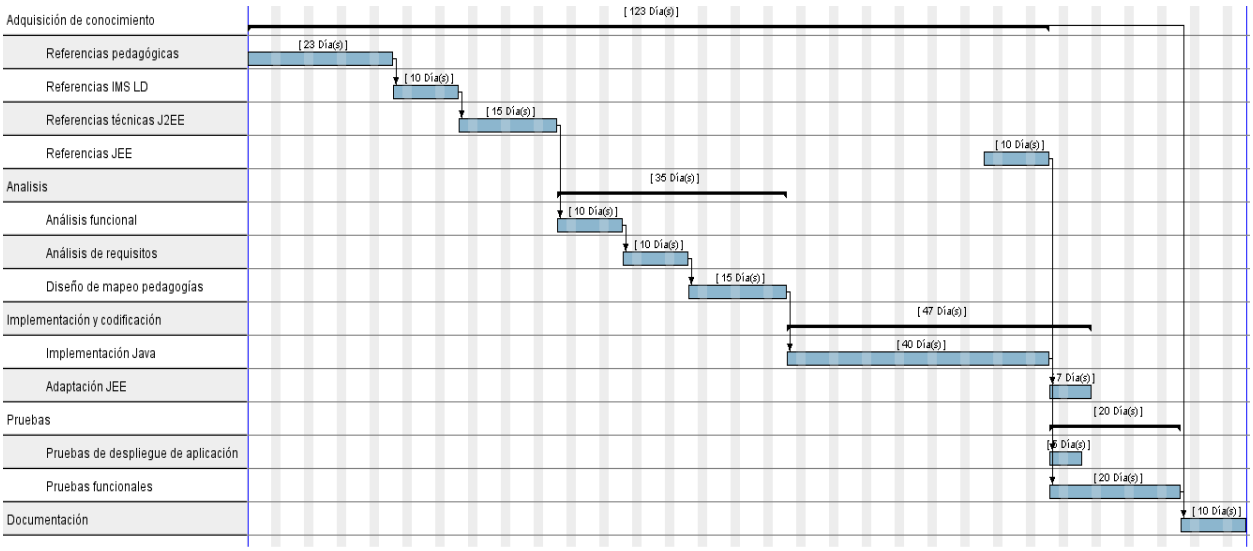


Figura 7-1. Diagrama de Gantt para el proyecto

7.3. COSTES DIRECTOS

En la siguiente tabla se muestra el resumen de la dedicación de recursos al proyecto desglosado por tareas y perfiles:

Tarea	Dedicacion total (jornadas)	Dedicacion Ing. Senior (jornadas)	Dedicación Analista programador (jornadas)
Referencias pedagógicas	23	10%	90%
Referencias IMSLD	10	20%	80%
Referencias J2EE	15	15%	85%
Referencias JEE	10	5%	95%
Analisis funcional	10	15%	85%
Analisis de requisitos	10	10%	90%
Diseño mapeo pedagogías	15	40%	60%
Implementación Java	40	0%	100%
Adaptación JEE	7	0%	100%
Pruebas despliegue	5	0%	100%
Pruebas funcionales	20	0%	100%
Documentación	10	20%	80%
TOTAL	**175**	**17,55**	**157,45**

Para el cálculo del precio por jornada y por perfil se ha tenido en cuenta una ponderación basada en las tarifas de varias consultoras para los perfiles de Analista Programador Java e Ingeniero/Consultor Senior durante el año 2010. El coste total de los recursos humanos dedicados al proyecto se muestra en la siguiente tabla:

	Jornadas	Tarifa jornada	Totales
Ingeniero Senior (Tutor)	17,55	360,00 €	6.318,00 €
Analista Programador (Alumno)	157,45	180,00 €	28.341,00 €
		Total	**34.659,00 €**

Dado que el software utilizado para el desarrollo del proyecto es de libre distribución, los únicos materiales a tener en cuenta en el cálculo de costes son dos ordenadores portátiles cuyo coste estimado se muestra en la siguiente tabla.

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable
Portatil Desarrollo (*Toshiba Tecra S5 Core 2 Duo T7500 2.2 GHz. 4GB RAM. 250GB. Windows XP Professional*)	1.400,00	90	8	60	168,00
Portatil Ing. Senior (*Dell Latitude E5510 Intel Core i3-370M Dual-Core 2.4GHz. 2GB RAM. 160GB; Intel GMA. Windows XP Professional*)	700,00	5	8	60	4,67
				Total	**172,67**

Para calcular este coste, se ha empleado la siguiente fórmula de de amortización:

[d] Fórmula de cálculo de la Amortización:

$$\frac{A}{B} x C x D$$

A = nº de meses desde la fecha de facturación en que el equipo es utilizado
B = periodo de depreciación (60 meses)
C = coste del equipo (sin IVA)
D = % del uso que se dedica al proyecto (habitualmente 100%)

El total de costes directos del proyecto asciende a: **34.831,67 €**

7.4. COSTES INDIRECTOS

Los costes indirectos por estiman a priori y consideran los costes derivados de gestión y seguimiento del proyecto. Se calcularán considerando un porcentaje estimado en un 20% de los costes directos.

El total de costes indirectos del proyecto asciende a: **6966,33 €**

7.5. CUADRO RESUMEN DEL PRESUPUESTO

Total del presupuesto del proyecto	
Total de costes directos	34.832 €
Total de costes indirectos	6.966 €
Resultado TOTAL del presupuesto	**41.798 €**

8. REFERENCIAS

[1] Eva Martínez Caro. *E-learning: Un análisis desde el punto de vista del alumno*. Universidad Politécnica de Cartagena (España), 2008. Desde: <http://www.utpl.edu.ec/ried/images/pdfs/volumen11N2/elearningunanalisis.pdf>

[2] Luisa Gavilán Arribas. *E-learning para personas sordas o con discapacidad auditiva*. Curso de Experto Profesional en E-learning 2.0. Curso 2007/08. Desde: <http://nadieduerma.com/wp-content/uploads/2008/07/lg_formacion_disc-auditiva14.pdf>

[3] Ing. Jorge Buabud, Ing. Rosana Hadad Salomón, Ing. Uriel Cukierman*. Impacto de la implementación del campus virtual del la UTN-FRT.* Facultad Regional Tucumán (Argentina), 2009. Desde: <http://www.tucuman.gov.ar/fraternidad/doc/IMPACTO DE LA IMPLEMENTACION DEL CAMPUS VIRTUAL DE LA UTN-FR.doc>

[4] Juan Manuel Fuentes. *E-learning ocupacional*. Revista eLearning América Latina. Noviembre 2009. Desde: < http://www.elearningamericalatina.com/edicion/agosto2/na_2.php>

[5] U.S. Government © Advanced Distributed Learning. *SCORM*. Desde: <http://www.adlnet.gov/Technologies/scorm/default.aspx>

[6] The Dublin Core® Metadata Initiative. *DublinCore.* Seoul, Korea. Desde: <http://dublincore.org/>

[7] IMS Global Learning Consortium, Inc. (2003). *IMS Learning Design Information Model 1.0.* <http://www.imsglobal.org>

[8] Neumann, S., Oberhuemer, P.: *Bridging the Divide in Language and Approach between Pedagogy and Programming: The Case of IMS Learning Design*. In: 15th International Conference of the Association for Learning Technology, Leeds (2008)

[9] Julia Minguillon, Enric Mor. *Personalización del proceso de aprendizaje usando learning objects reutilizables.* Universitat Oberta de Catalunya. Desde: <http://spdece.uah.es/papers/Minguillon_Final.pdf>

[10] IMS Global Learning Consortium, Inc. (2003). *IMS Learner Information Packaging.* <http://www.imsglobal.org/profiles/index.html>

[11] IMS Global Learning Consortium, Inc. (2003). *IMS Question and Test Interoperability.* <http://www.imsglobal.org/question/index.html>

[12] IMS Global Learning Consortium, Inc. (2003) *IMS Simple Sequencing.* <http://www.imsglobal.org/simplesequencing/index.html>

[13] IMS Global Learning Consortium, Inc. (2003). *IMS Content Packaging.* <http://www.imsglobal.org/content/packaging/index.html>

[14] Pilar Lacasa, Ana Belén García Varela y Héctor del Castillo. *Aprendizaje y desarrollo humano desde un enfoque sociocultural.* Capítulo 18. Universidad de Alcalá de Henares, 2004.

[15] Ing. Ignacio Chávez Arcega. *Conductismo, Cognitivismo y Diseño Instruccional.* Instituto Tecnológico de Tepic. Febrero de 2009., Desde: <http://www.virtualeduca.info/ponencias/381/Conductismo,%20Cognitivismo%20y%20Dise%F1o%20Instruccional.pdf>

[16] Ana Isabel Molina, Francisco Jurado, Ignacio de la Cruz, Miguel Ángel Redondo, and Manuel Ortega. *Tools to Support the Design, Execution and Visualization of Instructional Designs.* Dept. of Information Technologies and Systems, Computer Science and Engineering Faculty, Castilla - La Mancha University, 2008. Desde: <http://www.springerlink.com/content/c7l475j633873851/>

[17] Wily, D. *Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy*, AIT/AECT, The Instructional Use of Learning Objects, Association for Instructional Technology, 1-35, año 2002.

[18] Timothy Koschmann. *The Physiological and the Social in the Psychologies of Dewey and Thorndike: The Matter of Habit. Dept. of Medical Education*, Southern Illinois University. Año 2000. Desde: <http://www.umich.edu/~icls/proceedings/pdf/Koschmann.pdf>

[19] Arenas Vega, Cecilia. *El cognitivismo y el constructivismo.* Agosto 2006. Desde: <http://www.monografias.com/trabajos14/cognitivismo/cognitivismo.shtml>

[20] W. Palomino. *Teoría del aprendizaje significativo de David Ausubel*. Agosto 2006. Desde: <http://www.monografias.com/trabajos6/apsi/apsi.shtml>

[21] Díaz-Barriga, F. y Hernández, G. *Estrategia Docentes para un aprendizaje significativo (2ª Edición).* Ciudad de México, Editorial McGraw-Hill Interamericana, 2002.

[22] Dr. Pere Marquès*. Concepciones sobre el aprendizaje.* Noviembre 2006. Desde: <http://dewey.uab.es/pmarques/aprendiz.htm>

[23] C. George Boeree. *Teorías de la personalidad: B.F. Skinner.* Año 1998. Desde: <http://www.psicologia-online.com/ebooks/personalidad/skinner.htm>

[24] Catherine Zavalla, Malka Sepúlveda, Germán Passi. *El condicionamiento operante de B.F.Skinner.* Año 2008. Desde: <http://www.monografias.com/trabajos15/condic-skinner/condic-skinner.shtml>

[25] Manuel Francisco Aguilar Tamayo. *El mapa conceptual y la teoría sociocultural.* Universidad Autónoma del Estado de Morelos, México. Desde: <http://cmc.ihmc.us/cmc2006Papers/cmc2006-p80.pdf>

[26] Flórez Ochoa, Rafael. *Evaluación Pedagógica y Cognición.* McGraw-Hill Interamericana S.A., Bogotá, 1999.

[27] Claudio Tascón Trujillo. *Aportaciones de Bruner.* Universidad de Las Palmas de Gran Canaria, 2005. Desde: <http://www.ctascon.com/Aportaciones%20de%20Bruner.pdf>

[28] Mª Luz Rodríguez Palmero. *La teoría del aprendizaje significativo.* Centro de Educación a Distancia Santa Cruz de Tenerife. Año 2004. Desde: <http://cmc.ihmc.us/papers/cmc2004-290.pdf>

[29] Novak, J.D., 1982. *Teoría y Práctica de la Educación.* Alianza Universidad: Madrid.

[30] Hernandez-Leo. *Computacional Representation of Collaborative Learning Patterns using IMS Learning Design.* Educational Technology & Society, Vol. 8, No. 4, pp. 75-89, año 2005

[31] Dillenbourg, P. *Collaborative learning: Cognitive and computacional approaches.* Oxford, UK: Elsevier Science, 1999.

[32] Luz María Zañartu Correa. *Aprendizaje colaborativo: una nueva forma de Diálogo Interpersonal y en Red*. Contexto Educativo, 2000. Desde: <http://contexto-educativo.com.ar/2003/4/nota-02.htm>

[33] Eugenio Garza Sada. *Las Técnicas Didácticas en el Modelo Educativo del TEC de Monterrey.* Col. Tecnológico, Monterrey, Septiembre 2000. Desde: <http://www.sistema.itesm.mx/va/dide/publicaciones/inf-doc/tecnicas-modelo.PDF>

[34] VVAA. *El proceso de enseñanza y aprendizaje.* Curso de Aptitud Pedagógica. Instituto de ciencias de la educación de la Universidad Complutense, 2004.

[35] VVAA. *Didáctica de las matemáticas.* Curso de Aptitud Pedagógica. Instituto de ciencias de la educación de la Universidad Complutense, 2004.

[36] Lic. José Ben Hur Saravia. *Internet y Aprendizaje colaborativo.* Universidad Pedagógica Nacional Francisco Morazán De Honduras, 2001. Desde: <http://www.fepade.org.sv/cra/Saravia/Internet y Aprendizaje colaborativo.doc>

[37] Steve Booth-Butterfield. *Attribution Theory.* Septiembre, 1996. Desde <http://www.as.wvu.edu/~sbb/comm221/chapters/attrib.htm>

[38] Renato Vairo Belhot, João H. Lopes Guerra, Nídia Pavan Kuri. *Problem-based learning in engineering education.* Escola de Engenharia de São Carlos - USP, 1998. Desde <http://www.ineer.org/Events/ICEE1998/Icee/papers/251.pdf>

[39] Rob Koper. *Modeling units of study from a pedagogical perspective: the pedagogical meta-model behind EML.* Open University of the Nederlands, Jun 2001.

[40] Daniel Burgos and Rob Koper. *Practical pedagogical uses of IMS Learning Design's Level B.* Educational Technology Expertise Centre (OTEC). Open University of the Netherlands, octubre 2005. <http://dspace.ou.nl/handle/1820/471>

[41] Daniel Burgos Solans. *Estudio de la estructura y del comportamiento de las comunidades virtuales de aprendizaje no formal sobre estandarización del e-learning.* Universidad Europea de Madrid. Abril 2006. Recuperado desde: <http://dspace.ou.nl/bitstream/1820/626/1/BURGOSDaniel_e-Thesis_20-04-06.pdf>

[42] Van Es, René. Koper, Rob. *Testing the pedagogical expressiveness of LD.* Open University of the Netherlands Educational Technology Expertise Centre, febrero 2005. <http://dspace.ou.nl/handle/1820/305>

[43] Burgos, Daniel. *Extension of the IMS Learning Design Specification based on Adaptation and Integration of Units of Learning.* Learning 2008 93-17.

[44] Koper, R., Spoelstra, H., Burgos, D.. *Learning Networks using Learning Design. A first collection of papers.* Educational Technology Expertise Centre, The Open University of the Netherlands, noviembre 2004. <http://dspace.learningnetworks.org/handle/1820/291>

[45] IMS Global Learning Consortium, Inc. (2003). *IMS Learning Design Best Practice and Implementation Guide*. <http://www.imsglobal.org>

[46] Griffiths, D., Blat, J., García, R., Sayago, S.. *La aportación de recursos pedagógicos reutilizables.* Universitat Pompeu Fabra (Barcelona). <http://spdece.uah.es/papers/Griffiths_Final.pdf>

[47] Koper, R., & Olivier, B.. *Representing the Learning Design of Units of Learning.* Educational Technology & Society,7 (3), 97-111. Año 2004.

[48] Publications from Mizoguchi Lab. Department of Knowledge Systems, ISIR - Osaka University. Desde: <http://www.ei.sanken.osaka-u.ac.jp/pub/all-publications.html>

[49] Seiji Isotani, Akiko Inaba, Mitsuru Ikeda and Riichiro Mizoguchi. *An Ontology Engineering Approach to the Realization of Theory-Driven Group Formation.* Department of Knowledge Systems, ISIR - Osaka University. Desde: <http://www.ei.sanken.osaka-u.ac.jp/pub/isotani/isotani_CSCLJournal_draft.pdf>

[50] Sergio Gutiérrez Santos, Abelardo Pardo, Carlos Delgado Kloos. *Authoring Courses with Rich Adaptive Sequencing for IMS Learning Design*. J. UCS 14(17): 2819-2839. Año 2008.

[51] Villasclaras Fernández, E.D., Hernández Gonzalo, J.A., Hernández Leo, D., Asensio Pérez, J.I., Dimitriadis, Y., Martínez Monés, A. *InstanceCollage: A Tool for the Particularization of Collaborative IMS-LD Scripts Special Issue on New Directions in Advanced Learning Technologies.* Educational Technology & Society. 12(4):56-70, Octubre 2009.

[52] Davinia Hernández Leo, Juan I. Asensio Pérez, Yannis A. Dimitriadis. *IMS Learning Design Support for the Formalization of Collaborative Learning Patterns.* Department of Signal Theory,

Communications and Telematics Engineering. University of Valladolid, septiembre 2004. <http://dspace.ou.nl/handle/1820/233>

[53] Daniel Burgos, Colin Tattersall y Rob Koper. *¿Puede IMS Learning Design ser utilizada para modelar juegos educativos?* Open University of The Netherlands, mayo 2005. <http://dspace.ou.nl/handle/1820/367>

[54] Burgos, D., Tattersall, C., & Koper, R. (2007). *How to represent adaptation in eLearning with IMS Learning Design.* Interactive Learning Environments, 15(2), 161-170.

[55] R. Sosa Sánchez-Cortés, A. García Manso, J. Sánchez Allende, P. Moreno Díaz, A. J Reinoso Peinado. *B-Learning y Teoría del Aprendizaje Constructivista en las Disciplinas Informáticas: Un esquema de ejemplo a aplicar.* Año 2005. Desde: <http://www.formatex.org/micte2005/AprendizajeConstructivista.pdf>

[56] RELOAD. *Reusable eLearning Object Authoring & Delivery.* University of Bolton y University of Strathclyde. Disponible en <http://www.reload.ac.uk/>

[57] Open Universiteit Nederland (OUNL). *Coppercore v3.1.* <http://coppercore.sourceforge.net>

[58] *CopperAuthor.* En SourceForge.net desde Dic 1, 2004. <http://copperauthor.sourceforge.net>

[59] *SLeD. Service Based Learning Design System (2004).* Service Based Learning Design System Project. Desde: <http://sled.open.ac.uk>

[60] Tattersal et al. *IMS Learning Design Frequently Asked Question.* Año 2003. Desde: <http://learningnetworks.org/downloads/IMSLD/IMS Learning Design FAQ 1.0.pdf>

[61] IMS Global Learning Consortium, Inc. (2003). *IMS Learning Design XML Binding.* <http://www.imsglobal.org>

[62] Martin Weller (2006) *The SLeD project: Investigating Learning Design and Services, JISC E-learning Focus.* Desde: <http://www.elearning.ac.uk/features/sledproject>

[63] *UNFOLD.* UNFOLD Project, 2004. Disponible en <http://www.unfold-project.net>

[64] *Prolix GLM.* Desde: <http://sourceforge.net/projects/openglm/files>

[65] OUNL. *Learning Network for Learning Design.* Heerlen: Open University of The Netherlands, OTEC, 2004. Desde: <http://moodle.learningnetworks.org>

[66] Technologies Cogigraph inc © 2008 *.MOT y MOT+.* Desde: <http://www.cogigraph.com/Produits/MOTetMOTplus/tabid/995/language/en-US/Default.aspx>

[67] LAMS International Pty Ltd. Desde: <http://www.lamsinternational.com>

[68] Consortium Board TENCompetence. *ReCourse Learning Design Editor*. Desde: <http://www.tencompetence.org/ldauthor>

[69] Eclipse Foundation. *Eclipse IDE.* Desde <http://www.eclipse.org>

[70] JBoss Community. *JBoss Application Server.* Desde: <http://www.jboss.org/jbossas>

[71] Pardo, Abelardo, Abelardo Pardo, and Carlos Delgado Kloos. *Experiences with GRAIL: Learning Design support in .LRN.* Learning. Citeseer, 2007. 1-7.

[72] *.LRN Virtual Machine.* Gradient Lab de la Universidad Carlos III de Madrid (UC3M). Desde: <http://gradient.it.uc3m.es/pub/dotlrn_vmware_1_4.zip>

[73] Universitat Pompeu Fabra. Grupo de Tecnologías Interactivas. *Learning Technology Open Source - IMS Doc Validator.* <http://gti.upf.edu/leteos/newnavs/docvalidator.html>

[74] *Tiles Framework.* The Apache Software Foundation. <http://tiles.apache.org/>

[75] *MySQL AB.* Sun Microsystems, Inc. 2008-2009. Desde <http://www.mysql.com>

[76] *MySQL Documentation.* Sun Microsystems, Inc. 2008-2009. Desde <http://dev.mysql.com/doc>

[77] Paul DuBois. *Writing JDBC Applications with MySQL.* Desde <http://www.kitebird.com/articles/jdbc.html>

[78] *Hibernate.* Gavin King. <http://www.hibernate.org>

[79] *Spring Framework.* SpringSource. <http://www.springsource.org>

[80] *XML Tutorial.* W3Schools. <http://www.w3schools.com/xml/default.asp>

[81] Computer Home (TM). *HTML color codes and names.* Desde: <http://www.computerhope.com/htmcolor.htm>

[82] *JDOM*. The JDOM Project Foundation. Desde <http://www.jdom.org>

[83] Govind Seshadri. *Exploring the MVC design pattern.* <http://www.javaworld.com/javaworld/jw-12-1999/jw-12-ssj-jspmvc.html>

[84] *Apache Struts.* The Apache Software Foundation. Desde <http://struts.apache.org>

[85] Daniel Burgos, Noaa Barak, Vicente Romero (ATOS), Christian Glahn, Marcus Specht, Marion R. Gruber, Dominique Verpoorten, Sebastian Kelle, Roland Klemke (OUNL), Maurice Hendrix, Alexandra Cristea (Warwick). *Extensions and modifications of learning specifications and LMSs focused on adaptive learning.* Enero-2011. Desde: <http://www.grapple-project.org/public-files/deliverables/D5.3c-WP5-LearningSpecifications-v1.0.pdf>

[86] Sharples, P., Griffiths, D., & Scott, W. (2008). *Using Widgets to Provide Portable Services for IMS Learning Design.* Proceedings of the 5th TENCompetence Open Workshop "Stimulating Personal Development and Knowledge Sharing". Sofia, Bulgarian.

[87] Dai Griffiths (CETIS, UK). *Delivering flexible services for IMS Learning Design using widgets: achievements, limitations and prospects.* Seminar on "Learning Design" (Madrid, 2011).

[88] De La Fuente Valentín, L., Pardo, A., & Delgado Kloos, C. (2008). *Change is Good. Improving Learning Design Flexibility at Run-Time.* International Conference on Advanced Learning Technologies (pp. 1048-1050).

[89] *Apache Tomcat.* The Apache Software Foundation. Desde <http://tomcat.apache.org>

[90] *Jboss Tools for Eclipse.* Red Hat Middleware, LLC. Desde <http://www.jboss.org>

[91] Daniel Burgos and Rob Koper. *Practical pedagogical uses of IMS Learning Design's Level B.* Educational Technology Expertise Centre (OTEC). Open University of the Netherlands, octubre 2005. <http://dspace.ou.nl/handle/1820/471>

[92] Berlanga, A. J., García, F. J., Carabias, J. *IMS Learning Design: Hacia la Descripción Estandarizada de los Procesos de Enseñanza.* In M. Ortega Cantero (Ed.). Simposio Nacional de Tecnologías de la Información y de las Comunicaciones en la Educación, SINTICE'2005 (Granada, Spain, September 13-16). Madrid, Spain: Thomson. 2005. 95-102.

[93] *SQLyog.* Webyog Softworks Pvt. Ltd. 2009. Desde: <http://webyog.com/en>

[94] *Mozilla Firefox.* Mozilla Europe y Mozilla Foundation. Desde: <http://www.mozilla-europe.org/es/firefox/>

Apéndice A. MANUAL DE INSTALACIÓN

En éste manual se explica como desplegar la aplicación objeto de este Proyecto Fin de Carrera (PedaLea). Se dividirá esta explicación en los siguientes subapartados:

1. Breve descripción del entorno de desarrollo Eclipse / JBoss Tools / Tomcat.
2. Entorno necesario para el despliegue
3. Instalación de **PedaLea** "Paso a Paso".

1. Breve descripción del entorno de desarrollo Eclipse / JBoss Tools / Tomcat.

Eclipse [69] es una de las más conocidas herramientas de código abierto para el desarrollo de software. Eclipse comenzó como un proyecto de IBM Canadá en el año 2001 y desde entonces ha sido usado como Entorno Integrado de Desarrollo (en inglés IDE) para la mayoría de proyectos Java y J2EE.

JBoss Tools [90] es un conjunto de extensiones para Eclipse diseñadas especialmente para facilitar el desarrollo y despliegue de aplicaciones J2EE sobre JBoss.

Apache Tomcat [89] es básicamente un contenedor de servlets desarrollado por la Apache Software Foundation. Da soporte para la ejecución de Servlets y Java Server Pages (JSP).

Una opción para el despliegue de una aplicación web en Tomcat es a partir de un fichero en **formato WAR**. Este tipo de archivo es básicamente un fichero comprimido que contiene las carpetas de la aplicación web. Evita muchas complicaciones a la hora de realizar el despliegue de una aplicación web porque contiene la estructura de carpetas incluidas las librerías adicionales a utilizar por parte de la aplicación. Eclipse proporciona este tipo de archivo.

En el entorno de desarrollo **Eclipse y Tomcat** se conectan a través de las JBoss Tools para las tareas de arranque y parada de la aplicación, depuración y visualización de trazas.

2. Entorno necesario para el despliegue

Para poder desplegar la aplicación es necesario disponer de los siguientes elementos en el entorno del servidor:

- Instalación de Java(TM) SE Runtime Environment (build 1.6.0_07-b06)
- Instalación de Apache Tomcat 6.0.18
- Instalación de MySQL Server 5.0
- Disponer de la última versión del archivo WAR de la aplicación PedaLea.

La ventaja de utilizar archivos de despliegue WAR y la arquitectura elegida es que no es necesario configurar más librerías para el correcto despliegue de la aplicación.

3. Instalación de PedaLea "Paso a Paso"

El primer paso es la ejecución del script de instalación de la base de datos "PFC.sql". Este archivo se encuentra dentro del archivo WAR que contiene la aplicación:

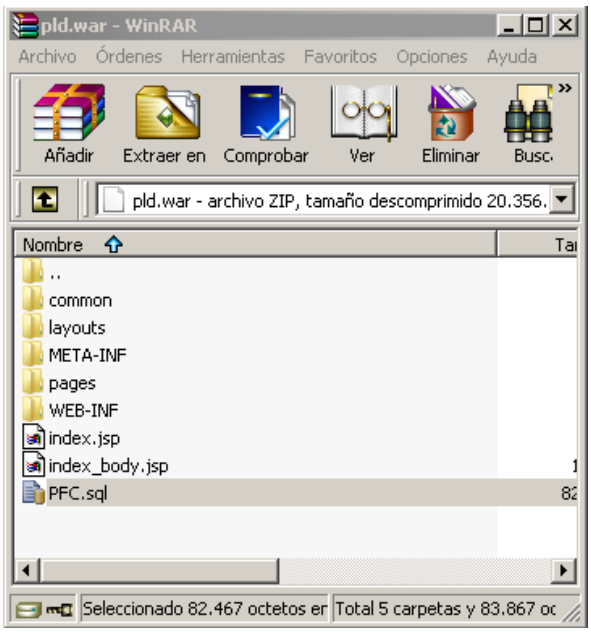


Figura Ap-1. Estructura de directorios del archivo WAR de la aplicación

La ejecución de este script se debe hacer mediante algún software que permita la conexión con el motor de base de datos (por ejemplo SQLyog [93]). Una vez ejecutado el script se habrá creado en la base de datos un nuevo conjunto de tablas con unos datos de muestra para la aplicación web:

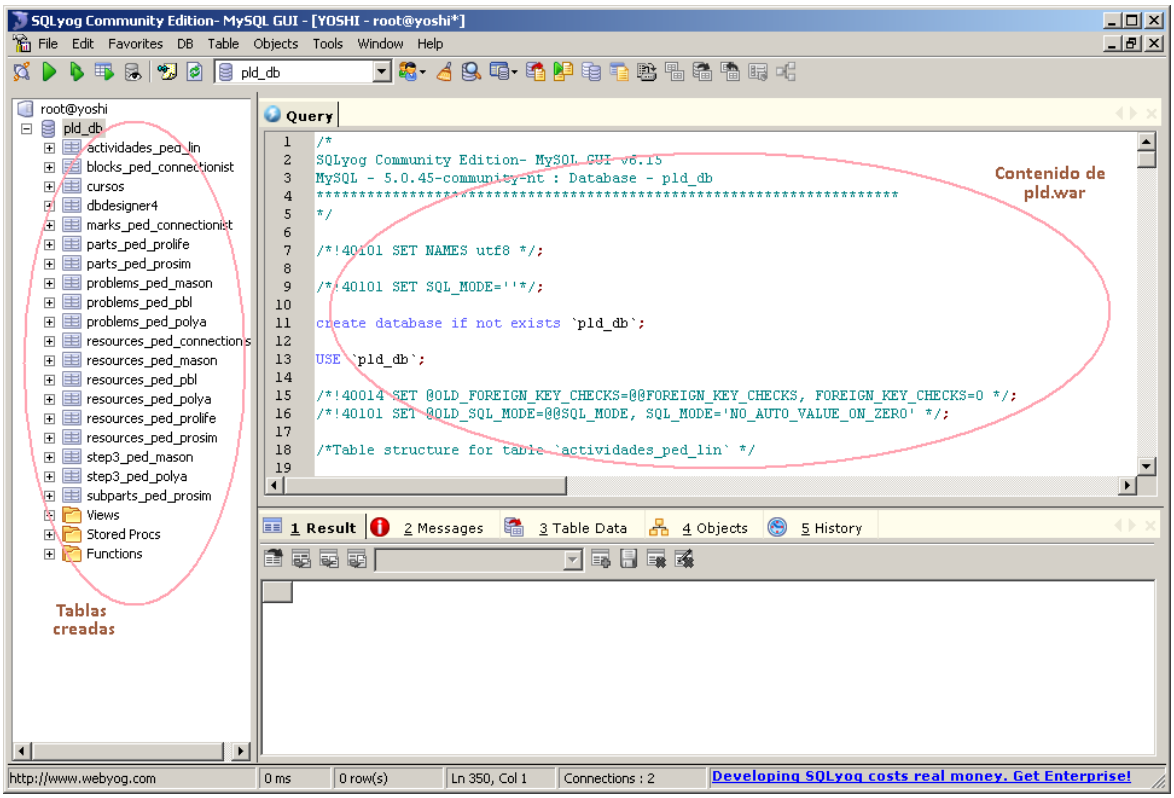


Figura Ap-2. Ejecución del script de creación de la base de datos.

El siguiente paso es el despliegue del archivo WAR en Tomcat. Para ello, accederemos a la sección de *manager* dentro de Tomcat (típicamente http://serverName:8080/manager/html) y a la sección de "Archivo WAR a desplegar":

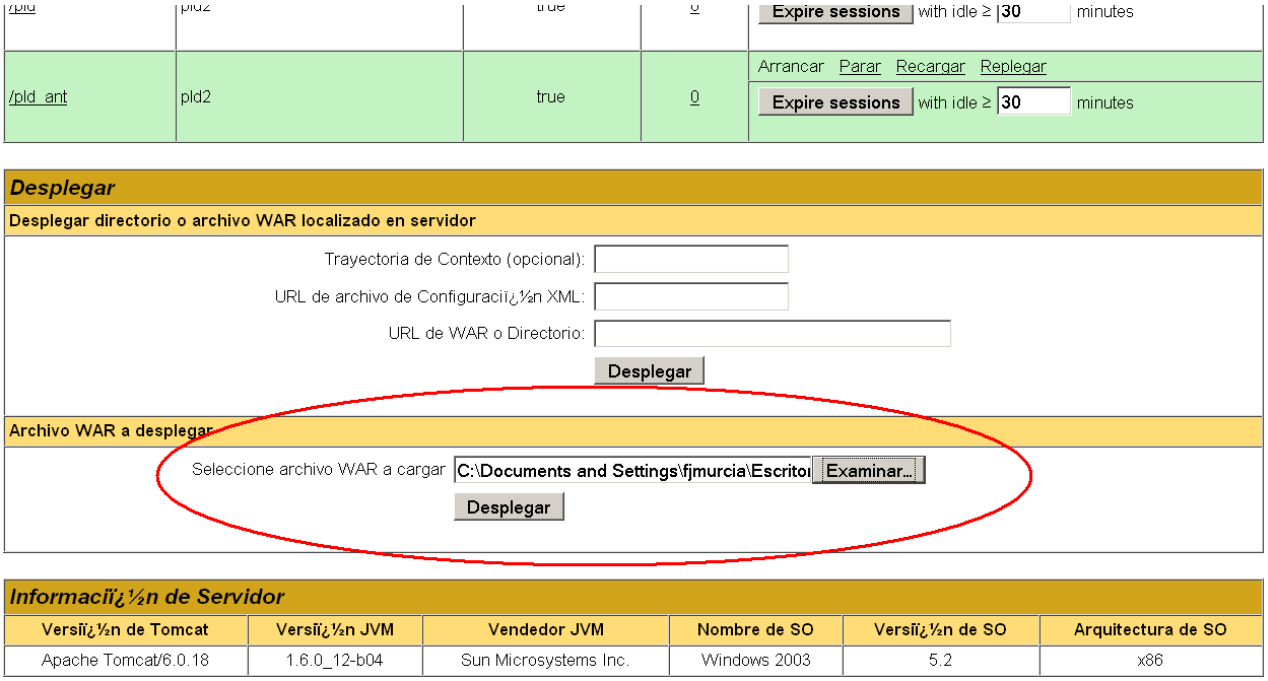


Figura Ap-3. Vista de *manager* para el envío del archivo WAR hacia Tomcat

Una vez enviado el archivo WAR al servidor, es necesario dejar pasar un tiempo prudencial de aproximadamente 1 minuto para que se ejecute por completo del despliegue de la aplicación. Una vez transcurrido ese tiempo la aplicación estará ya disponible y operativa.

Apéndice B. MANUAL DE USUARIO

En este apartado se recoge el manual de usuario para la aplicación desarrollada para este Proyecto Fin de Carrera (**PedaLea**). Para interactuar con esta aplicación web será necesario el uso de un navegador web. Si bien otros navegadores son compatibles, se recomienda utilizar Mozilla Firefox [94] y una resolución de pantalla de al menos 1280 píxeles de ancho.

Consideraciones iniciales

PedaLea es una aplicación web que implementa una herramienta de autor para la creación de cursos de aprendizaje a distancia basados en el estándar IMS LD con una cierta orientación pedagógica.

El perfil de usuario de PedaLea es el de un docente que desea crear un curso para impartirlo online empleando una cierta pedagogía. El docente elegirá un tipo de pedagogía o metodología para aplicar en el desarrollo del curso e interactuará con la aplicación web, dando de alta bloques, temas o evaluaciones y añadiendo recursos en función de las particularidades del curso que esté creando.

Visto a alto nivel, PedaLea es una interfaz entre los docentes e IMS LD, ya que los profesores interactúan con PedaLea en términos pedagógicos y PedaLea interactúa con IMS LD en términos del estándar.

El objetivo de la aplicación, es por lo tanto generar cursos completos basados en el estándar de IMS LD con una orientación pedagógica concreta mediante una interfaz sencilla.

Una vez creado el curso a impartir y añadidos los recursos pertinentes, el docente tendrá la posibilidad de guardar el curso en el repositorio para que esté disponible para futuras ediciones o bien descargarse desde la aplicación web un archivo comprimido (.zip) que contendrá tanto los ficheros propios de IMS LD como los recursos que el docente hubiera añadido. El docente podría cargar este fichero ZIP en una plataforma que soporte IMS LD y desplegar así el curso creado.

PedaLea ofrece diferentes pedagogías ya implementadas y una plataforma flexible que permitirá la adición de nuevas pedagogías o metodologías de aprendizaje.

Pantalla de inicio y menú de la aplicación

La disposición de los elementos en la pantalla de inicio presenta una cabecera, un menú a la izquierda, una parte principal y un pie de página. En la cabecera se encuentran los botones de selección de idioma, se mostrarán los textos en el idioma configurado en el navegador del cliente que accede a la aplicación entre inglés y castellano.

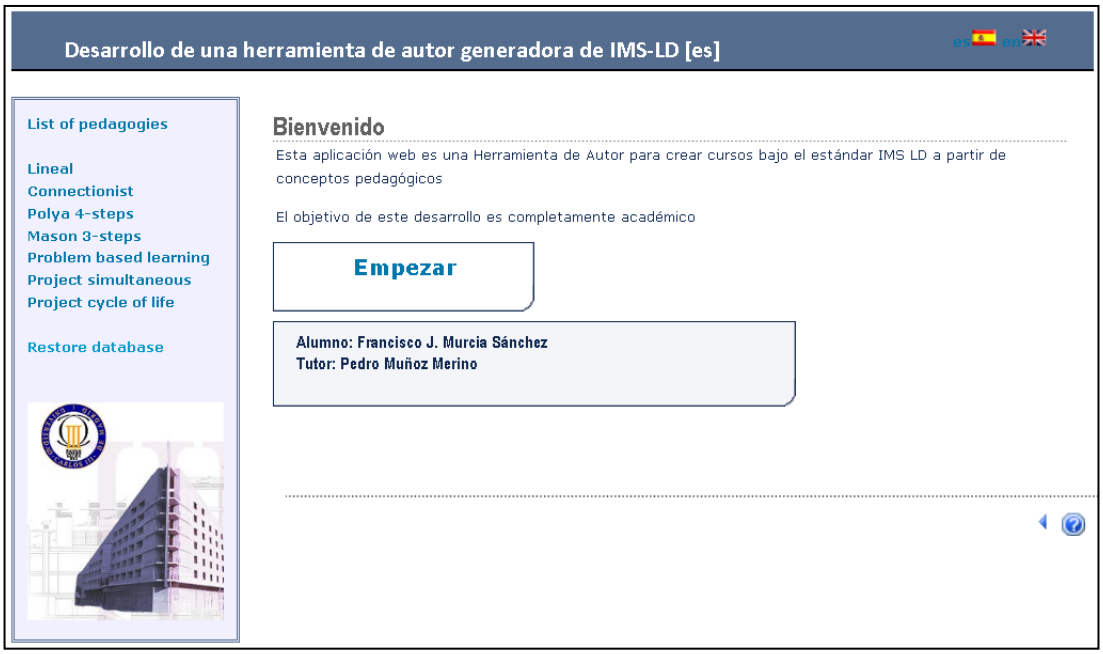


Figura Ap-4. Pantalla inicial de la aplicación.

El menú de la izquierda dispone de accesos directos para generar cursos con las diferentes orientaciones pedagógicas que ofrece la aplicación. Además, en el primer enlace del menú de la izquierda se puede acceder a todo el listado de pedagogías con una breve explicación de las mismas.

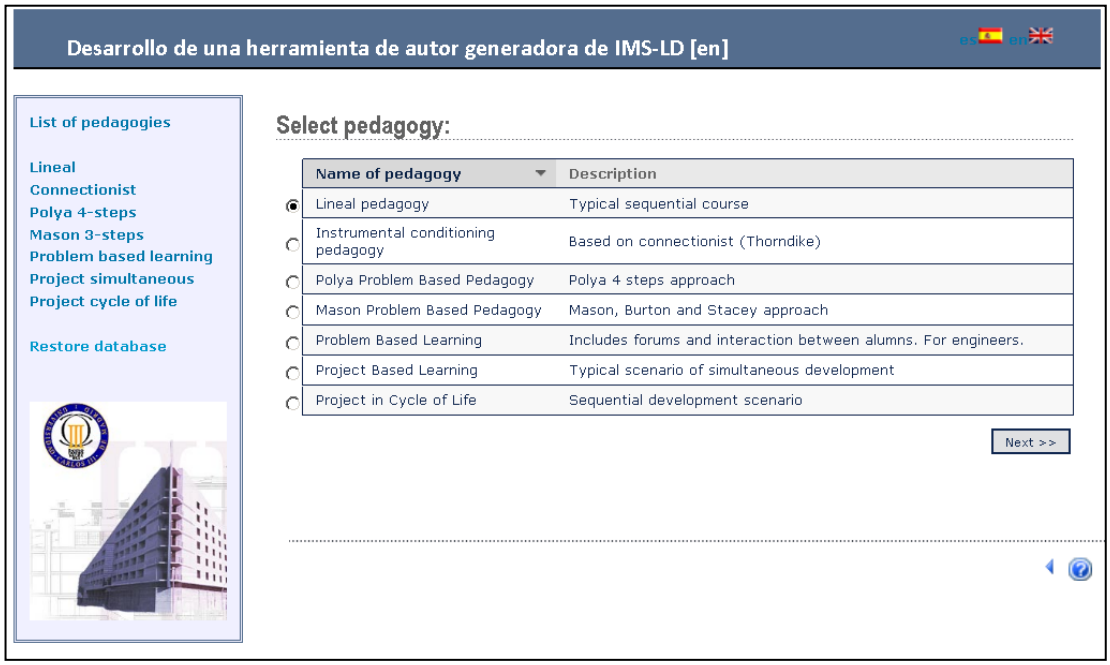


Figura Ap-5. Pantalla para la selección de pedagogía.

Al acceder a cualquier tipo de pedagogía, se llega a un listado de todos los cursos que hayan sido creados con esa orientación pedagógica, se muestra en la siguiente imagen:

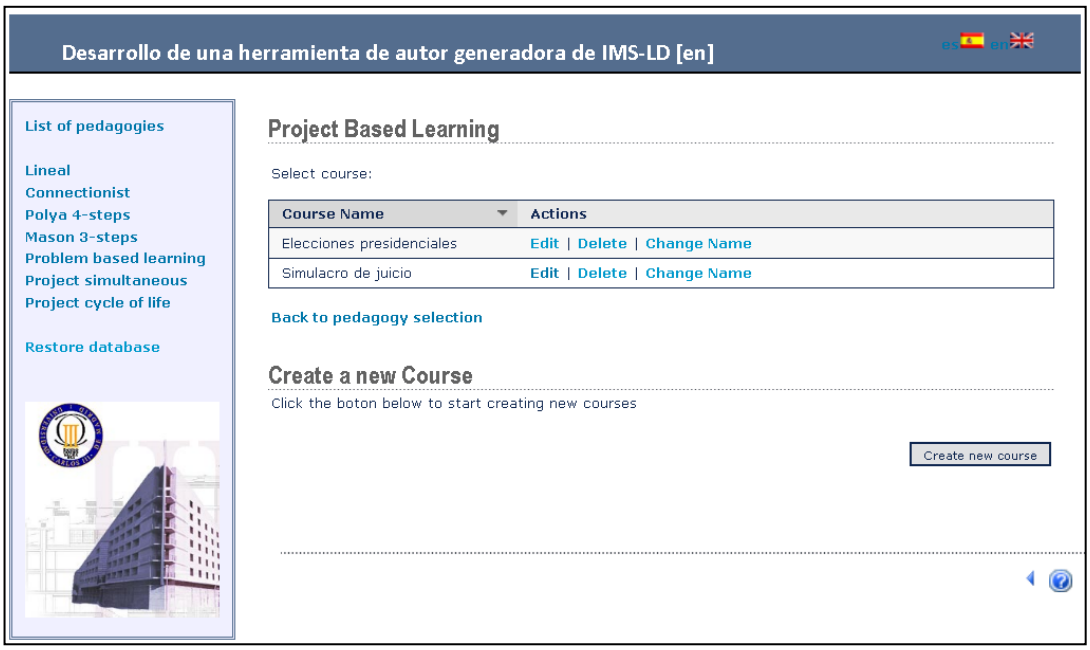


Figura Ap-6. Pantalla para la selección de un curso de una pedagogía concreta.

Para cada curso, se dispone de tres opciones:

- Editar > A través de esta opción se accede al curso. Una vez dentro de la edición de un curso, será posible realizar una edición del mismo o generar el archivo .zip para cargarlo en una plataforma compatible con IMS LD.
- Borrar > Se realizará el borrado del curso. Este borrado no tiene la opción de deshacerse.
- Cambiar nombre > Con esta opción se accede a la pantalla de cambio de nombre, que es igual para todas las pedagogías:

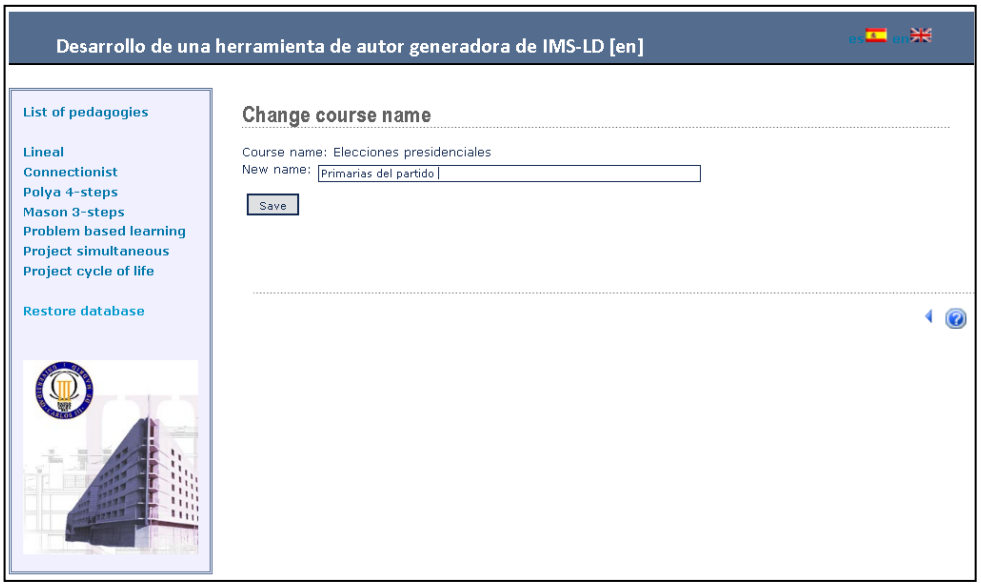


Figura Ap-7. Pantalla de edición para cambiar el nombre de un curso.

Desde esta pantalla de listado de cursos también se accede al botón de creación de un nuevo curso (botón abajo a la derecha). Esta pantalla de creación de nuevo curso es común a todas las pedagogías y es la siguiente:

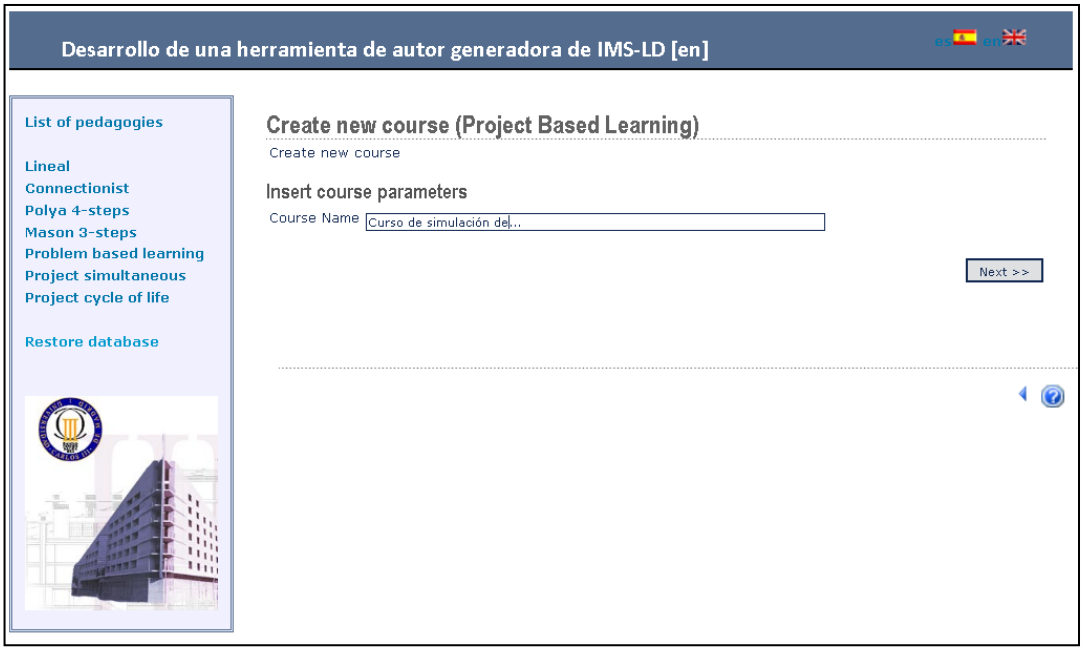


Figura Ap-8. Pantalla para la creación de un nuevo curso.

Simplemente se asigna un nombre al nuevo curso que se quiera crear y se hace click en el botón de siguiente. Una vez creado el curso, la aplicación directamente llevará al usuario a la pantalla de edición del curso recién creado.

Pedagogía Lineal

Se trata de una pedagogía en la que el profesor crea un curso para que sea ejecutado por parte de los alumnos de manera continuada, sin separación en bloques ni con la creación de roles específicos para cada tipo de alumno. No existirán elementos de *feedback* para el profesor como evaluaciones ni tampoco elementos de comunicación como foros de debate o similares.

Al acceder a la pedagogía lineal aparecerá el siguiente formulario para la edición del curso:

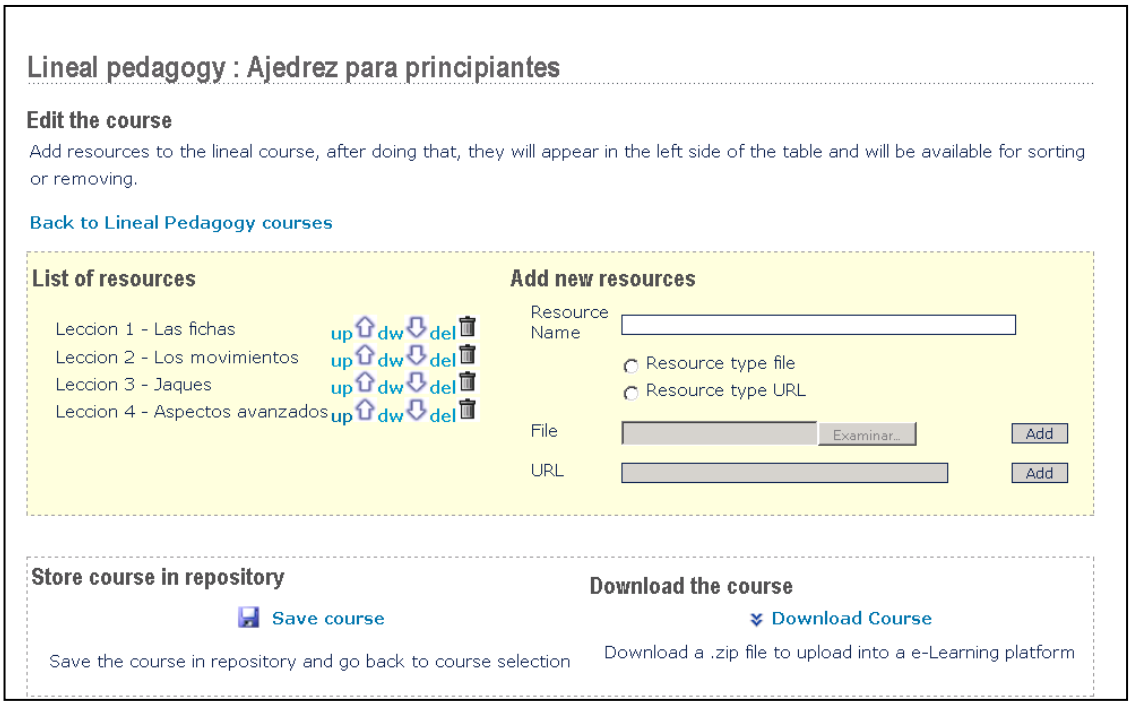


Figura Ap-9. Pantalla de edición para pedagogía lineal.

Existen dos áreas en la pantalla de edición del curso:

- Listado de recursos > En la parte izquierda de la pantalla se muestran los títulos de los recursos que ha ido incorporando el profesor al curso. El orden en el que se encuentran estos recursos será el mismo en el que se mostrarán a los alumnos en tiempo de ejecución en la plataforma online. El profesor dispone de botones para cambiar el orden de los recursos y también tiene la posibilidad de eliminarlos.
- Adición de recursos > En la parte derecha de la pantalla se muestra el formulario de adición de nuevos recursos. En este formulario, el profesor crea un título para el recurso y elige el tipo de recurso que añadirá que puede ser o bien un fichero o bien una URL. Al añadir un recurso nuevo, la pantalla se refrescará y se mostrará en el listado de la izquierda la nueva entrada añadida

Una vez terminada la adición de recursos y colocados en el orden deseado, el profesor podrá guardar el curso y volver al menú de selección de cursos para la pedagogía lineal o bien podrá descargarse el curso para cargarlo en una plataforma online.

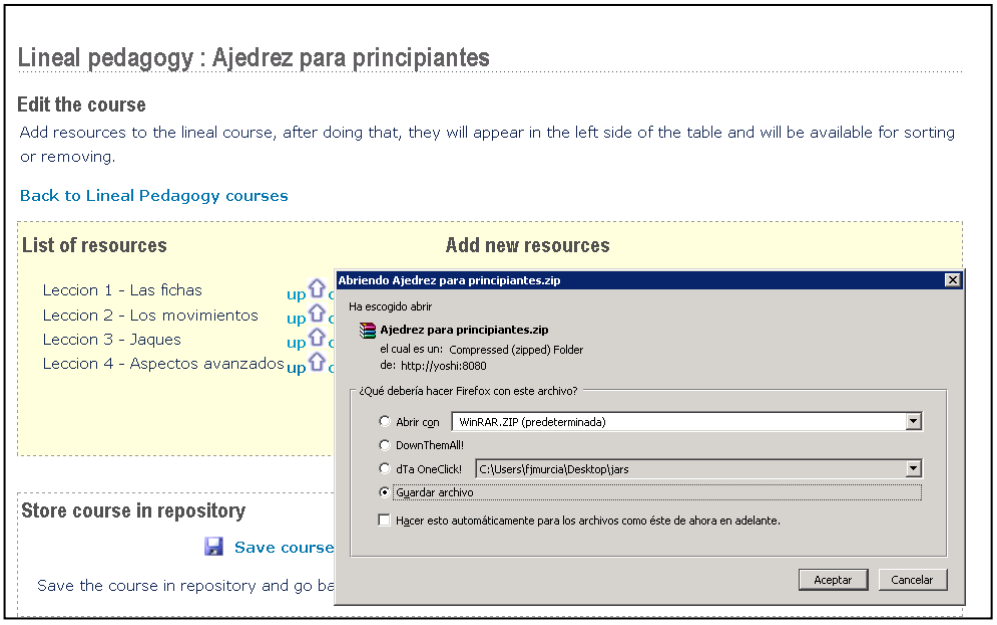


Figura Ap-10. Acceso para la descarga del curso a través de la acción "*Download course*".

El contenido del archivo comprimido es el siguiente:

- imsmanifest.xml que contiene el código IMS LD con la información del curso.
- ficheros subidos por el profesor

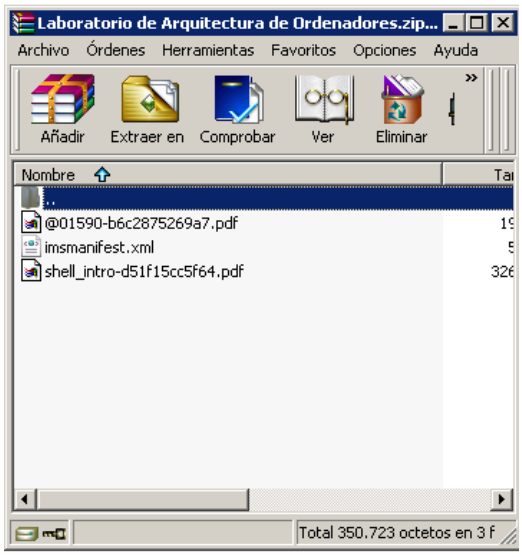


Figura Ap-11. Contenido del fichero comprimido generado por la aplicación

Pedagogía orientada a problemas. Aproximación de los 4 pasos de Polya

Esta pedagogía está basada en la técnica de resolución de problemas de Polya. Básicamente, esta pedagogía propone que para que el aprovechamiento de aprendizaje sea máximo en la resolución de problemas, el docente tiene que guiar al alumno hacia la solución en orden y con los siguientes pasos:

1. Comprensión del problema
2. Diseñar un plan para abordar el problema
3. Ejecución del plan
4. Revisión de los resultados

Durante el tiempo de ejecución, el alumno ejecutará un curso divido en tantos bloques como problemas haya añadido el docente en la fase de edición. Para cada problema, el alumno irá ejecutando linealmente cada una de los cuatro pasos de Polya hasta llegar a la solución del problema.

Una forma adecuada de crear un problema en base a los cuatro pasos de Polya sería entregar el siguiente tipo de material de formación por cada uno de los pasos:

1. Enunciado del problema. Se entregaría en este paso el enunciado completo del problema a resolver, el entorno que lo envuelve, la historia que lo precede... En resumen, se debería entregar al alumno en este punto el material oportuno para que sea capaz de asimilar qué es lo que se espera de la resolución de este problema.
2. Plan de actuación. En este segundo paso, se espera que el alumno sea capaz de diseñar un plan de actuación para abordar el problema.
3. Para la ejecución del plan el docente debería entregar al alumno una plantilla o modelo de la memoria que debería rellenar el alumno. Realmente en este caso es el alumno quien debería trabajar a su ritmo y posteriormente realizar la entrega de sus resultados. Para implementar este apartado, la aplicación proporciona al docente durante el tiempo de edición de una casilla para marcar si en este paso el alumno debe subir algún archivo o no.
4. Solución del problema. En este cuarto punto el profesor debería añadir la solución al problema para que sean los alumnos los que la contrasten con su solución y puedan de esa manera ser críticos con su trabajo y crear nuevas asociaciones de conceptos e incorporar nuevos métodos para resolver problemas.

Para proporcionar una interfaz sencilla en la creación de este tipo de cursos, la aplicación web proporciona la siguiente pantalla al acceder a la edición de un curso basado en la pedagogía de Polya:

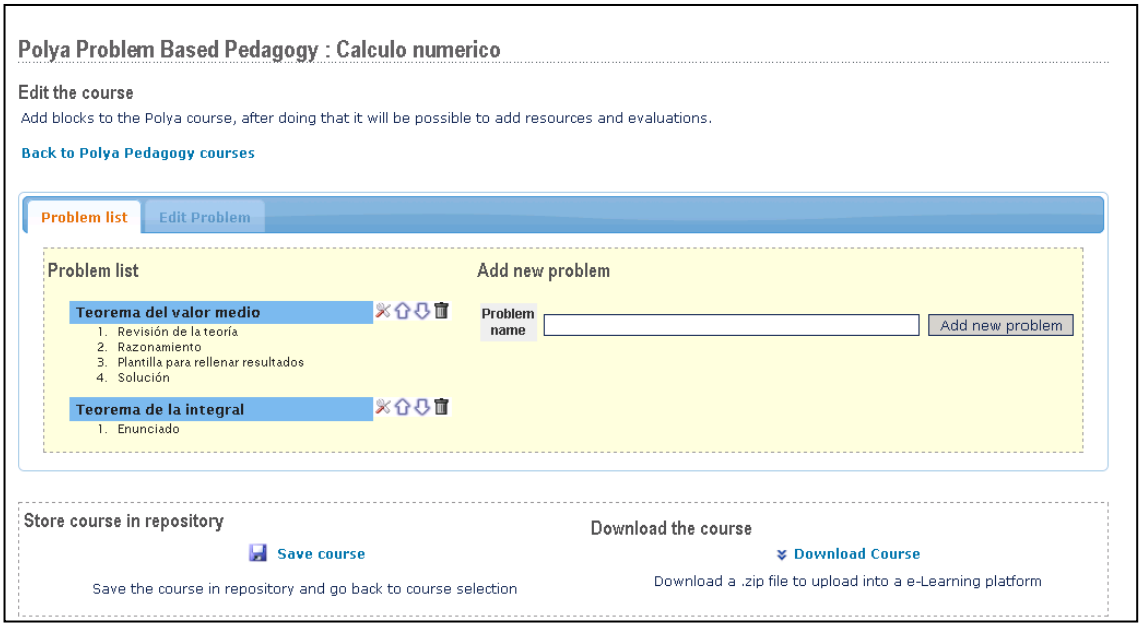


Figura Ap-12. Vista para la edición de un curso basado en los cuatro pasos de Polya

La pantalla de edición de este tipo de curso se presenta dividida en dos pestañas o *tabs*.

La pestaña de la izquierda estará inicialmente activa. Dentro de esta pestaña, en la zona de la izquierda se muestran los diferentes problemas creados y una enumeración de los pasos que lo componen. Cada problema tiene las siguientes opciones:

- Editar > Al acceder a la opción de editar, se activará el acceso a la sección de edición del problema.
- Subir > Para adelantar el orden de aparición del problema en tiempo de ejecución.
- Bajar > Para atrasar el orden de aparición del problema en tiempo de ejecución.
- Borrar > Elimina el problema y todos los pasos que contuviera. No es posible deshacer este borrado.

Por otro lado, dentro de la pestaña de listado de problemas, en la parte derecha se permite al docente agregar un nuevo problema, para lo que únicamente tendrá que introducir el nombre del nuevo problema y pulsar en el botón de añadir.

Al acceder a la opción de editar para cualquiera de los problemas del listado, se activará la pestaña de la derecha:

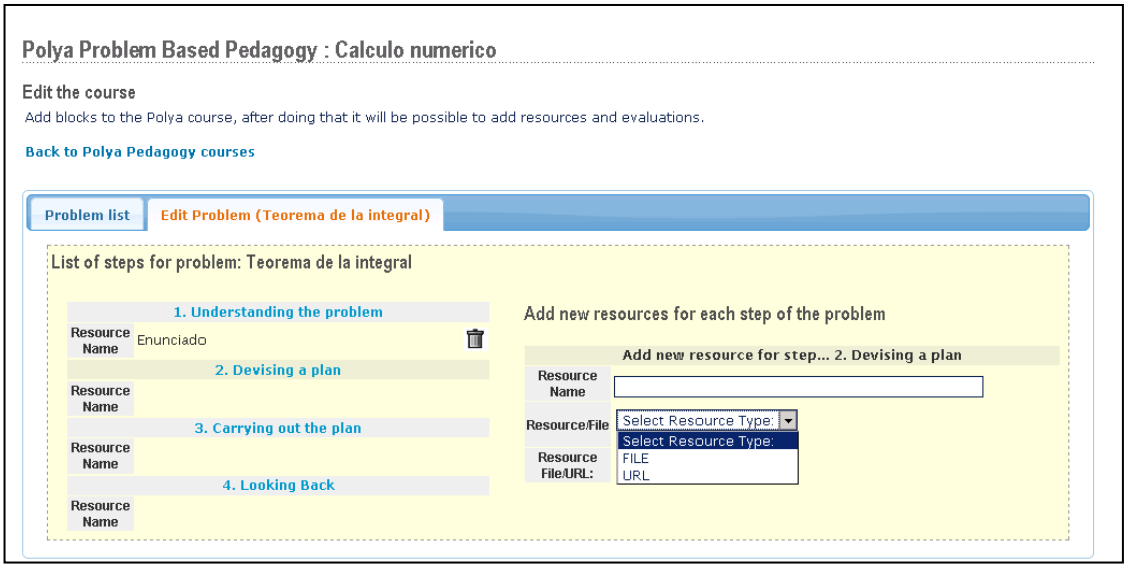


Figura Ap-13. Vista de edición de los diferentes pasos para la pedagogía basada en los cuatro pasos de Polya

Esta pantalla de la figura anterior, guía al docente para la inserción de un recurso fichero o URL para cada uno de los cuatro pasos de la pedagogía de Polya. Para este caso, no se contempla la posibilidad de cambiar el orden de los recursos ya que la pedagogía de Polya recoge que los cuatro pasos deben ser dados en el mismo orden.

Para el paso 3 en el que el alumno debe ejecutar el problema hay una variación, se muestra en la siguiente imagen:

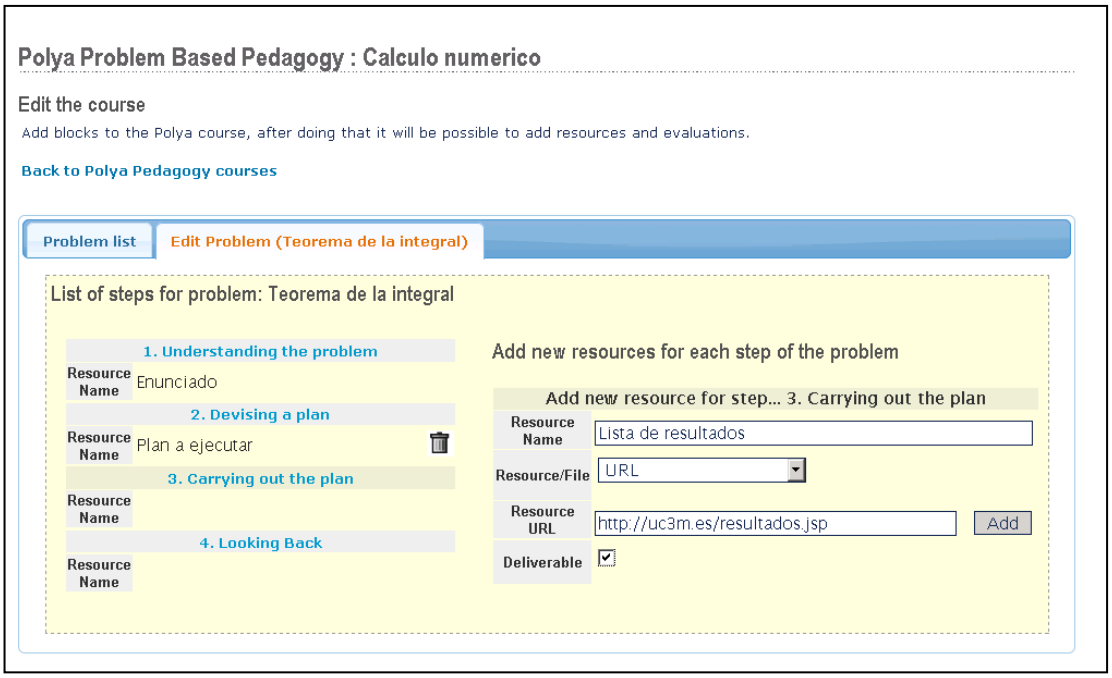


Figura Ap-14. Detalle del paso 3 para la pedagogía de Polya.

La imagen anterior corresponde al paso 3 en la que se puede observar la casilla de "Entregable" o "*Deliverable*", al activarla, el docente indica que durante el tiempo de ejecución el alumno tendrá que entregar un archivo al docente para su evaluación o supervisión. La entrega de este archivo se hace en tiempo de ejecución a través de un campo de formulario de tipo *upload file.*

Al igual que en la pedagogía lineal, el docente puede descargarse en cualquier momento el curso. En función de los diferentes parámetros introducidos en la edición, el archivo comprimido a obtener contendrá los siguientes archivos:

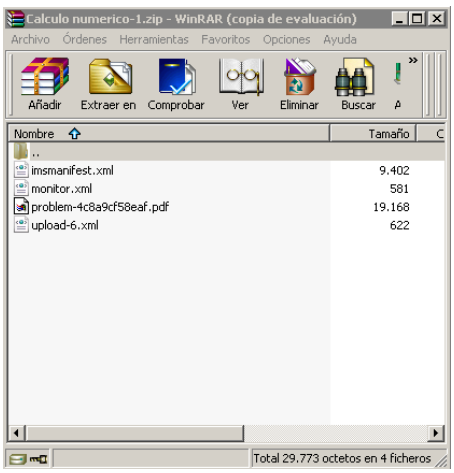
– imsmanifest.xml que contiene el código IMS LD con la información del curso. – ficheros subidos por el profesor – monitor.xml que contiene la información necesaria para que el docente pueda evaluar a los alumnos en tiempo de ejecución – upload-X.xml que contiene la información necesaria para que los alumnos puedan entregar sus archivos en tiempo de ejecución.	

Figura Ap-15. Contenido del fichero comprimido generado por la aplicación para la pedagogía de Polya

Pedagogía orientada a problemas. Aproximación de los 3 pasos de Mason, Burton and Stacey.

Al igual que en la pedagogía orientada a problemas de Polya, la aproximación de Mason, Buto y Stacey (en adelante simplemente Mason) propone una estructura de curso basada en la partición del curso en problemas, cada uno en un bloque diferente.

De la misma manera que en la pedagogía de los 4 pasos de Polya, los autores de esta pedagogía proponen que el aprovechamiento máximo del alumno sucede cuando se siguen para cada problema los siguientes tres pasos:

1. Entrada al problema.
2. Ataque del problema.
3. Revisión del resultado.

Durante el tiempo de ejecución, el alumno ejecutará un curso divido en tantos bloques como problemas haya añadido el docente en la fase de edición. Para cada problema, el alumno irá ejecutando linealmente cada uno de los tres pasos propios de esta metodología.

Una forma adecuada de crear un problema en base a los tres pasos de Mason sería entregar el siguiente tipo de material de formación por cada uno de los pasos:

1. Material con el planteamiento del problema. En este paso, el docente debería entregar a los alumnos un material que les permitiera contextualizar el problema y comprender a la perfección cuál es el objetivo u objetivos del mismo y cuál es el entorno para su resolución.
2. Material para la ejecución del mismo. En este paso el docente debería dotar al alumno de las herramientas necesarias para la resolución del problema. Este material puede ser alguna herramienta de software (ej. Un programa de simulación o de cálculo) , así como una plantilla para los resultados o similar. En este paso, el docente, durante el tiempo de creación del curso puede elegir si espera o no un material de vuelta por parte de los alumnos para la evaluación del ejercicio.
3. Solución del problema. En este paso se debería proporcionar al alumno la solución del problema y una guía para que pueda plantearse su propio resultado. De esta manera, según los autores de esta pedagogía, el alumno es capaz de incorporar nuevas formas de afrontar los problemas y ser crítico con los resultados obtenidos.

Cuando se accede en la aplicación a este tipo de pedagogía aparece la siguiente pantalla:

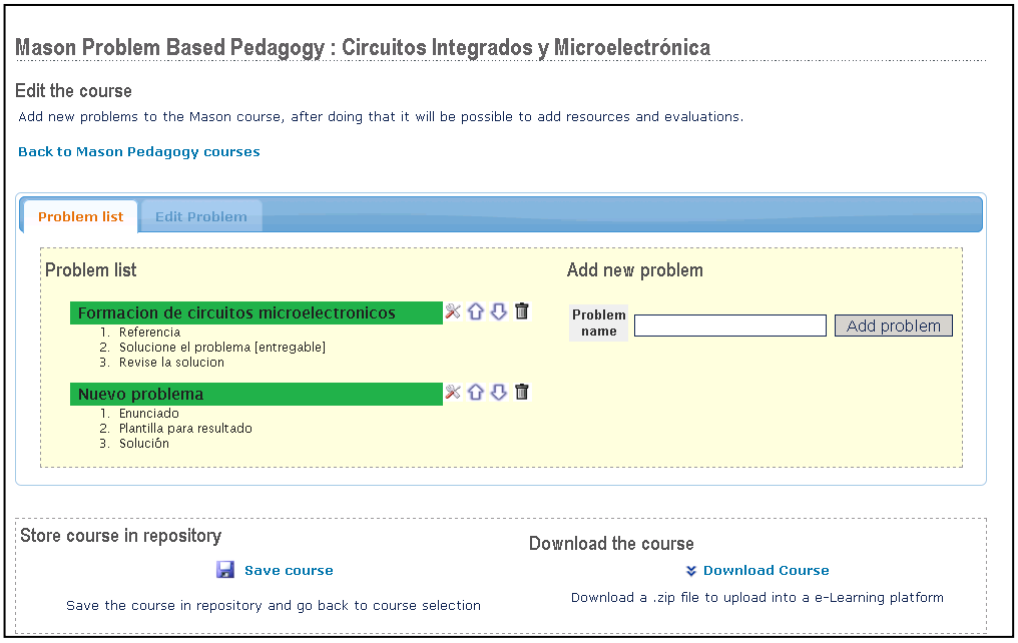


Figura Ap-16. Vista para la edición de un curso basado en los tres pasos de Mason

Como puede apreciarse, la pantalla se encuentra dividida en pestañas o *tabs* para facilitar a los docentes el trabajo de creación y edición de nuevos cursos.

En el *tab* de la izquierda aparecen dos bloques diferenciados: listado de problemas a la izquierda y adición de nuevo problema a la derecha. El listado de problemas muestra una serie de opciones para cada uno de los problemas:

- Subir > adelanta el orden de aparición de este problema en el tiempo de ejecución.
- Bajar > atrasa el orden de aparición de este problema en el tiempo de ejecución.
- Editar > con esta opción se activa la pestaña de la derecha en la que se cargarán los datos relacionados con el problema que se desea editar.
- Borrar > elimina el problema. Esta acción no se puede deshacer.

Cuando el docente accede a la opción de editar para cualquiera de los problemas, se activará el *tab* de la derecha con el siguiente contenido:

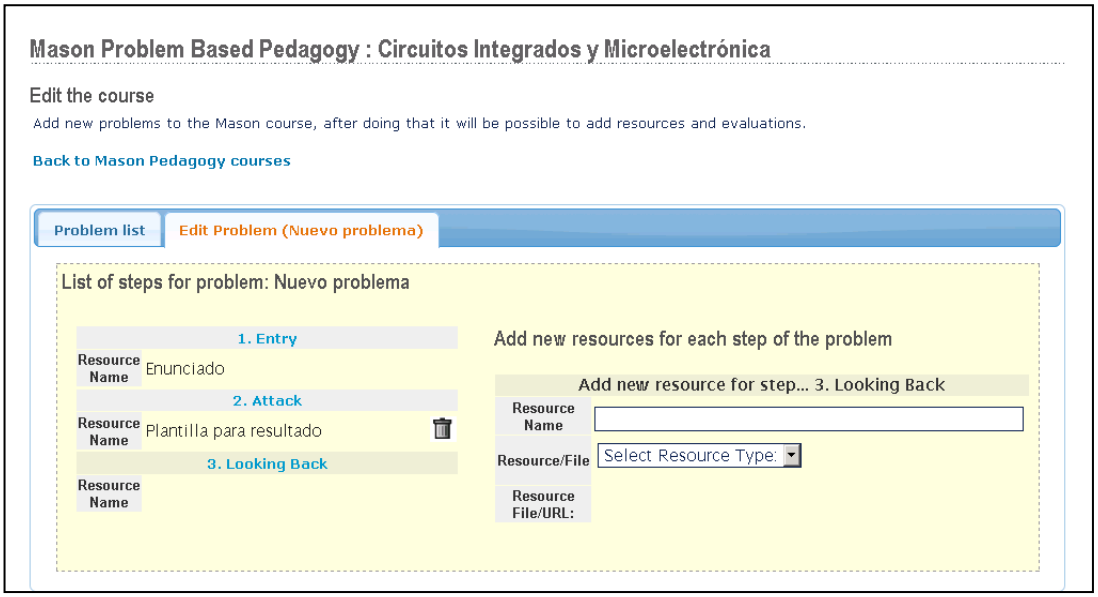


Figura Ap-17. Detalle para la edición de los recursos asociados a cada paso de la pedagogía de Mason

En esta pestaña de edición de problemas, el docente irá subiendo el material adecuado para la ejecución de cada uno de los pasos de esta pedagogía orientada a problemas. Al igual que en el caso de la pedagogía de Polya, en el paso de ejecución del problema, en este caso el paso 2, el docente podrá elegir si se requiere del alumno un fichero con la solución del problema marcando la casilla de "Entregable" o *"Deliverable"*.

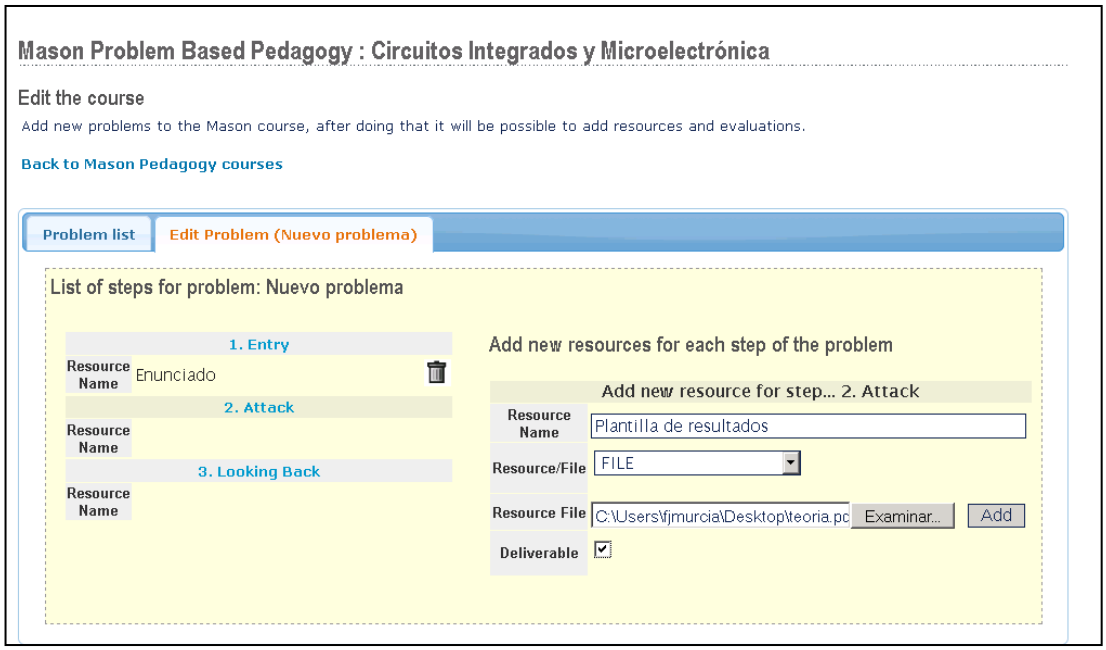


Figura Ap-18. Paso 2 de la pedagogía de Mason.

Una vez completado todos los pasos del problema, el docente podrá volver a la pestaña de adición de nuevo problema o modificar los contenidos de alguno de los problemas.

En cualquier momento durante la edición del curso, el docente puede descargar el curso en formato zip para cargarlo en una plataforma *online*. El contenido del fichero que obtendrá será similar al siguiente:

- imsmanifest.xml que contiene el código IMS LD con la información del curso.
- ficheros subidos por el profesor
- monitor.xml que contiene la información necesaria para que el docente pueda evaluar a los alumnos en tiempo de ejecución
- upload-X.xml que contiene la información necesaria para que los alumnos puedan entregar sus archivos en tiempo de ejecución.

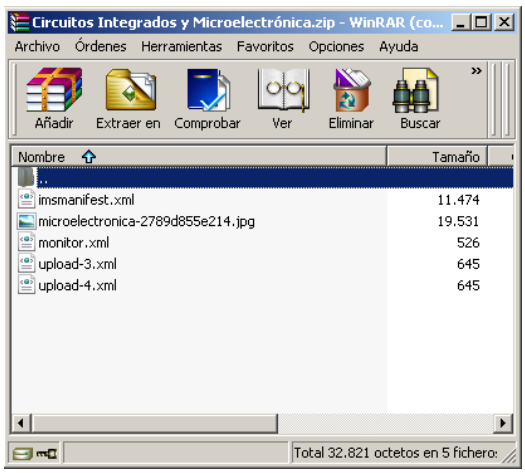


Figura Ap-19. Contenido del fichero comprimido generado por la aplicación para la pedagogía de Mason

Aprendizaje basado en problemas para ingenieros.

Esta pedagogía tiene su origen en el aprendizaje basado en problemas, con algunas adaptaciones para el caso en el que los alumnos son ingenieros [38]. Según esta metodología, un problema que debiera ser resuelto mediante la cooperación entre los alumnos siguiendo los siguientes pasos:

- Presentación del enunciado. Normalmente se tratará más de un escenario problemático que de un único problema aislado.
- Listado de aquello que se conoce. En este apartado, los estudiantes deberían compartir sus conocimientos y responder a la pregunta: *"¿Qué es lo que sabemos?"*
- Desarrollar un enunciado. Los alumnos en este caso deberían llegar a un enunciado común para afrontar el problema a resolver.
- Identificar los requisitos. Los alumnos deberían responder en este punto a la pregunta *"¿Qué es lo que necesitamos saber?"*
- Identificar las diferentes alternativas o hipótesis. En este punto, los alumnos deberían llegar a responder a la pregunta: *"¿Qué deberíamos hacer?"*
- Presentar y defender la solución.

Un curso basado en esta pedagogía, requiere de la interacción y comunicación de los alumnos entre ellos, para lo cual, el código IMS LD generado contendrá la definición de varios foros de debate, allí donde los alumnos tengan que llegar a un acuerdo común. De este modo, los alumnos podrán compartir sus conocimientos y sus descubrimientos, aportando todos al beneficio de hallar la solución al problema planteado.

PedaLea proporciona a los docentes una manera realmente sencilla de generar cursos bajo esta pedagogía porque únicamente tendrá que proporcionar por cada problema un enunciado y los recursos adicionales que considere necesarios para resolver el problema. Por ejemplo, si fuera un curso de programación y el problema fuera un bajo rendimiento de una aplicación, el profesor subiría como enunciado la descripción de la aplicación y del problema de rendimiento y como recursos adicionales podría subir el código de la aplicación y la documentación del análisis.

Cuando el docente accede a un curso basado en la pedagogía basada en problemas encontrará una pantalla como la siguiente:

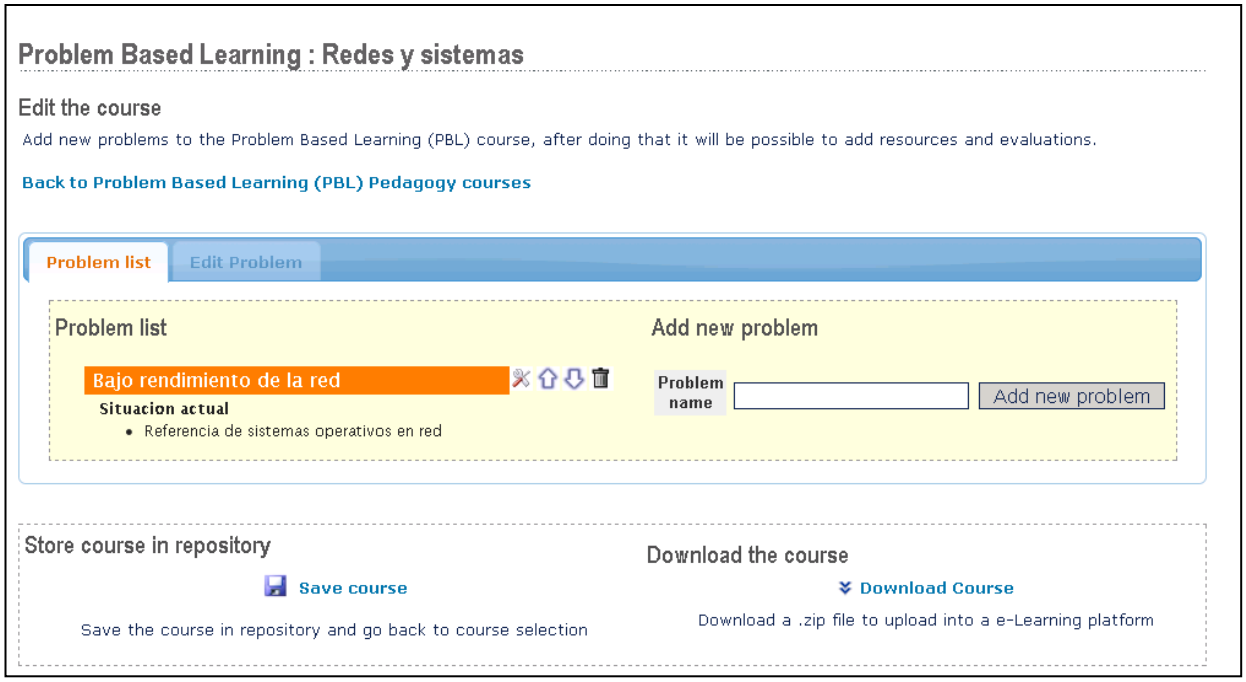


Figura Ap-20. Vista de la edición de cursos basados en PBL

La pantalla de inicio de edición se divide en dos pestañas. La pestaña que está activa inicialmente contiene los problemas añadidos para el curso y el formulario de adición de un nuevo problema. Por cada problema, el profesor cuenta con las siguientes opciones:

- Subir > adelanta el orden de aparición de este problema en el tiempo de ejecución.
- Bajar > atrasa el orden de aparición de este problema en el tiempo de ejecución.
- Editar > activa la pestaña de la derecha en la que se cargarán los datos relacionados con el problema que se desea editar.
- Borrar > elimina el problema. Esta acción no se puede deshacer.

Cuando se añade un nuevo problema o se accede a la opción de editar alguno de los existentes, la pestaña de la derecha se activará y se cargarán los datos del problema en cuestión que se quiere modificar o crear:

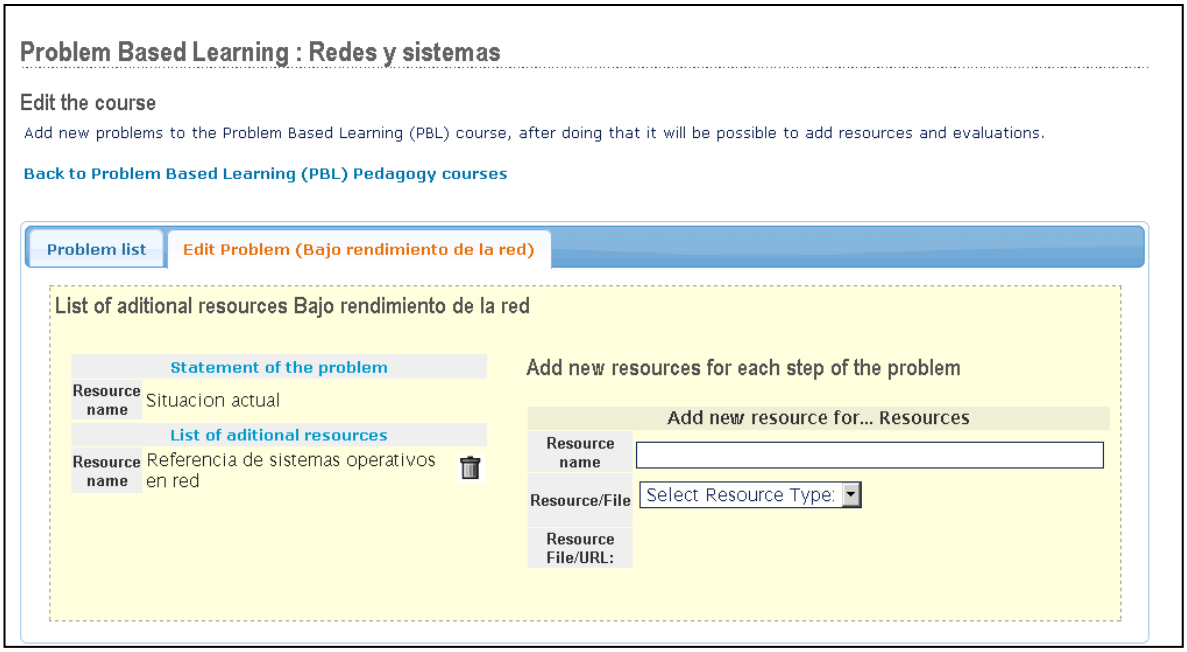


Figura Ap-21. Detalle para la adición de recursos en cursos basados en la pedagogía PBL.

El docente deberá añadir en esta pestaña un enunciado que explique la problemática a resolver y opcionalmente podrá añadir otro material a modo de documentación de referencia para la resolución del problema.

Una vez terminada la adición de problemas y recursos, el docente podrá descargar el ZIP del curso con el siguiente contenido:

- imsmanifest.xml que contiene el código IMS LD con la información del curso.
- ficheros subidos por el profesor
- monitor.xml que contiene la información necesaria para que el docente pueda evaluar a los alumnos en tiempo de ejecución
- upload-X.xml que contiene la información necesaria para que los alumnos puedan entregar sus archivos en tiempo de ejecución.
- res-stepX-PX.html que contiene un sencillo texto de invitación a los diferentes foros en los que se espera que participe el alumno
- step3-PX.xml y step3-PX_teacher.xml que contienen las propiedades para que los alumnos puedan crear un enunciado propio para el problema y permite al profesor monitorizar este progreso

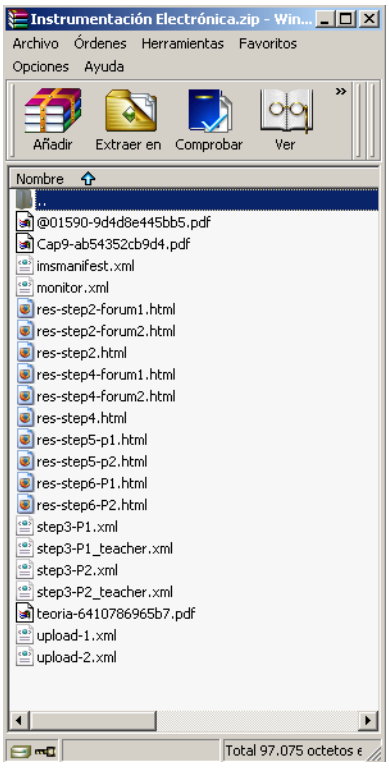


Figura Ap-22. Contenido del fichero comprimido generado por la aplicación para la pedagogía orientada a problemas (PBL)

Pedagogía orientada a proyectos en desarrollo simultáneo.

Esta pedagogía se centra en entornos de aprendizaje colaborativo [30] en los que cada alumno o grupo de alumnos asume un rol para llevar a cabo una parte de un proyecto conjunto.

Este tipo de pedagogía sería aplicable, por ejemplo, a un curso de estudiantes de derecho para los que el docente quiere proporcionar diferentes puntos de vista acerca de un delito. Si la educación fuera presencial, el docente separaría a los alumnos en diferentes grupos para que asumieran partes diferentes del proyecto.

Siguiendo con el ejemplo, un grupo de alumnos se encargaría de pensar y ejecutar las acciones policiales, otro grupo de las acciones judiciales y así con tantos grupos como puntos de vista quiera proporcionar el profesor. Además, dentro del propio grupo de trabajo, podrían existir subgrupos específicos que se encargaran de subpartes del proyecto. Una vez más, siguiendo con el ejemplo, el grupo de los alumnos que se encargan de la parte de acciones judiciales, se podría dividir en el subgrupo de abogados defensores, subgrupo de fiscales y subgrupo de jueces.

PedaLea proporciona soporte a este tipo de cursos a través de una sencilla interfaz que permite al docente centrarse en la tarea de describir el curso y evitando la complejidad tecnológica. En este caso, la pantalla principal está dividida en tres pestañas diferentes:

- Listado de partes y subpartes y formulario de adición de nuevos grupos de trabajo.
- Edición de las partes y adición de nuevas subpartes
- Edición del subgrupo y adición de recursos

La pestaña principal contiene el listado de partes y subpartes como se muestra a continuación:

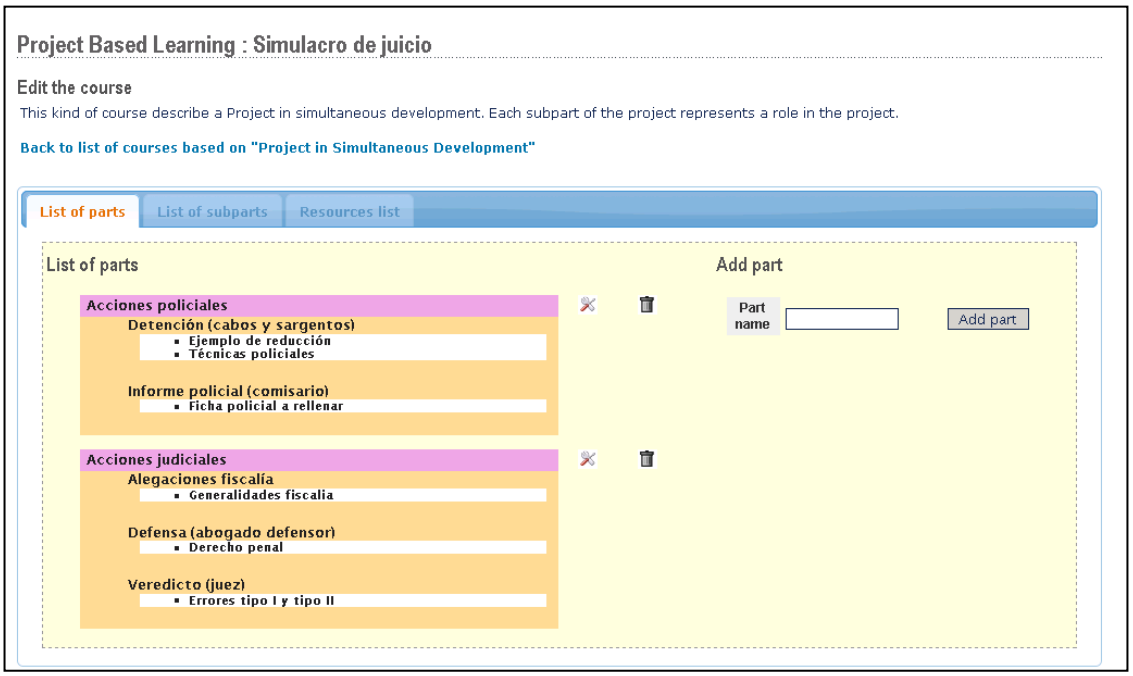


Figura Ap-23. Vista de la edición de cursos basados en proyectos en desarrollo simultáneo

El formulario de la derecha sirve al docente para ir añadiendo las partes que componen el proyecto. De este modo, cada una de las partes añadidas dispondría de las opciones de edición y de borrado. En este caso no es útil ordenar las diferentes partes ya que en tiempo de ejecución, los alumnos recibirán el material relacionado con la parte del proyecto en la que hayan sido asociados.

Cuando el docente accede a la opción de edición de una de las partes, la segunda pestaña se activará y permitirá acceder a la pantalla de edición de una parte del proyecto:

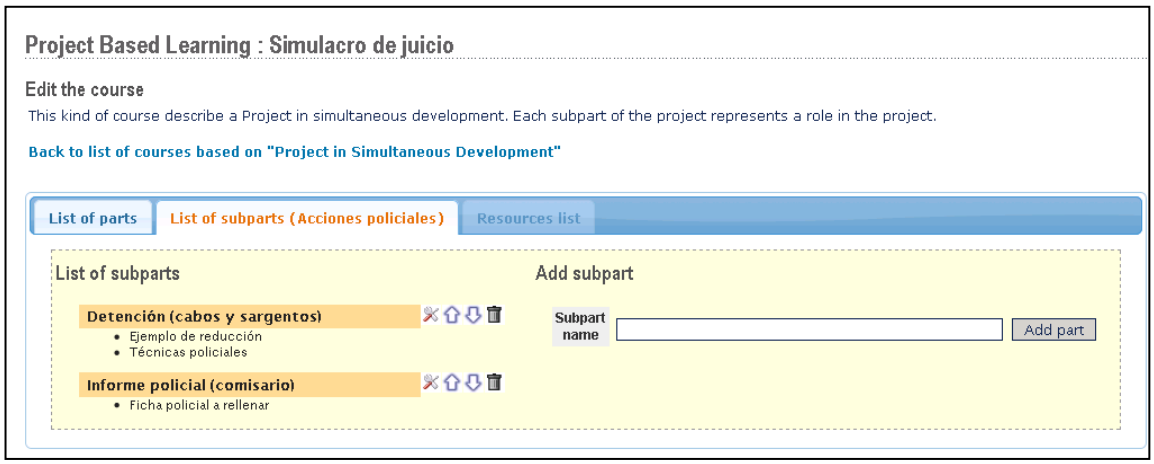


Figura Ap-24. Detalle de la vista para la edición de los recursos de cada fase en la pedagogía orientada a proyectos en desarrollo simultáneo

Esta pestaña permitirá al profesor añadir nuevas subpartes para la parte que está en edición mediante el formulario en la parte derecha.

Además, desde esta pestaña se puede consultar el listado de subpartes y acceder a las diferentes opciones que existen para cada una de ellas:

- Subir > en este caso tiene un significado estético porque los alumnos acabarán siendo asignados a una subparte de un proyecto. En el caso de que un proyecto se componga de alguna parte indivisible en subpartes, se añadirá una única subparte para que los alumnos sean asignados a ella.
- Bajar > como en el caso de subir, tiene un significado estético y no funcional.
- Editar > permite acceder a la pestaña de edición de la subparte para añadir recursos.
- Borrar > elimina la subparte y todos los recursos que tuviera asociados. Esta acción no se puede deshacer.

Cuando el docente acceda a la opción de editar una subparte, la tercera pestaña se mostrará:

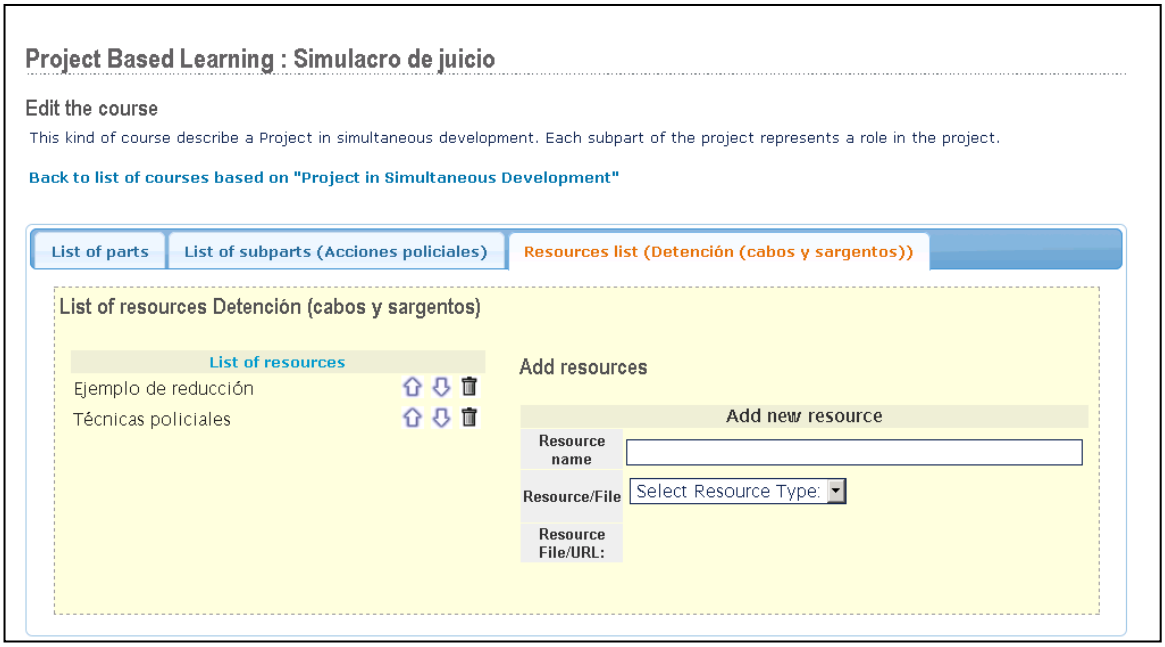


Figura Ap-25. Detalle de la vista para la edición de los recursos de cada fase en la pedagogía orientada a proyectos en desarrollo simultáneo

Desde esta pestaña, el profesor puede visualizar a la izquierda los recursos que están asignados a la subparte en proceso de edición y a la derecha dispondrá del formulario para poder añadir nuevos recursos. Para el conjunto de recursos, se presenta la opción de cambiarlos de orden mediante las acciones de *"subir"* y *"bajar"*. Igualmente se muestra la opción de eliminar un recurso.

Una vez concluida la edición del curso, el docente podrá descargar el archivo ZIP para cargar en la plataforma de educación. El contenido de este fichero será de la siguiente manera:

- imsmanifest.xml que contiene el código IMS LD con la información del curso relacionada con la asignación de roles para cada subparte.
- ficheros subidos por el profesor

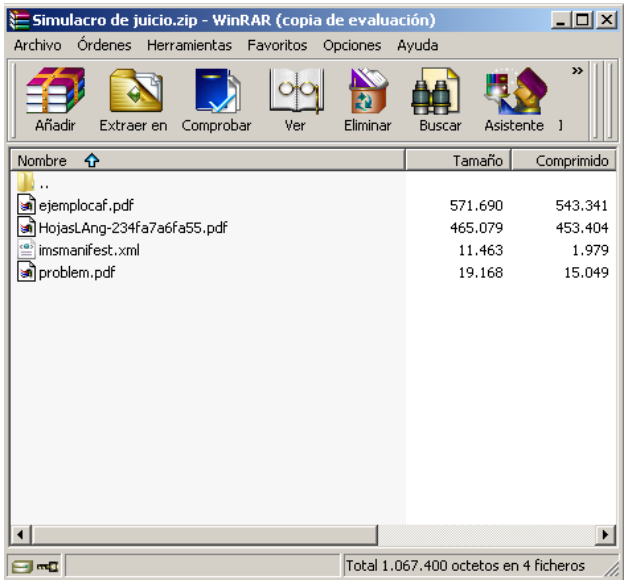


Figura Ap-26. Contenido del fichero comprimido generado por la aplicación para la pedagogía orientada a proyectos en desarrollo simultáneo

Pedagogía orientada a proyectos en desarrollo secuencial.

La principal diferencia de esta pedagogía con la anterior es que en este caso, los alumnos tendrán que esperar a la finalización de la anterior parte del proyecto para comenzar la suya. Por esto es por lo que en este caso en lugar de hablar de partes, se hablará de fases del proyecto.

En esta pedagogía se realiza un paso de información entre los líderes de cada una de las fases del proyecto y se establecen foros de comunicación para que los integrantes del proyecto puedan transmitirse sus dudas o aportaciones.

Un ejemplo de aplicación de este tipo de pedagogía sería un curso de programación en el que el docente quiere establecer diferentes fases para que los alumnos trabajen de manera conjunta. De esta manera, el proyecto podría contener una primera fase de análisis, una segunda fase de desarrollo y una última fase de pruebas. Se realizarían dos pases de información. El primero entre la fase de análisis y la fase de desarrollo, en la que típicamente los alumnos de la fase de análisis transferirían los documentos necesarios para que los integrantes de la fase de desarrollo pudieran empezar sus trabajos. El segundo pase de información es el que realizarían los alumnos que trabajen en la fase de desarrollo a los alumnos que hubieran sido asignados a la fase de pruebas, en este paso típicamente se transferirían los fuentes de la aplicación o directamente el ejecutable y un plan de pruebas (probablemente aportado inicialmente por los analistas).

PedaLea proporciona a los profesores que desearan adaptar esta pedagogía a sus cursos una interfaz sencilla en la que añadirían inicialmente las fases del proyecto y posteriormente los recursos que fueran necesarios para cada fase. Siguiendo con el ejemplo anterior, en la fase de análisis añadiría algunos recursos relacionados con los requisitos de la aplicación, en la fase de desarrollo algún manual y en la fase de pruebas alguna herramienta o lo aquello fuera oportuno para orientar y facilitar el trabajo de los integrantes de esta fase. La pantalla principal de la aplicación es la siguiente:

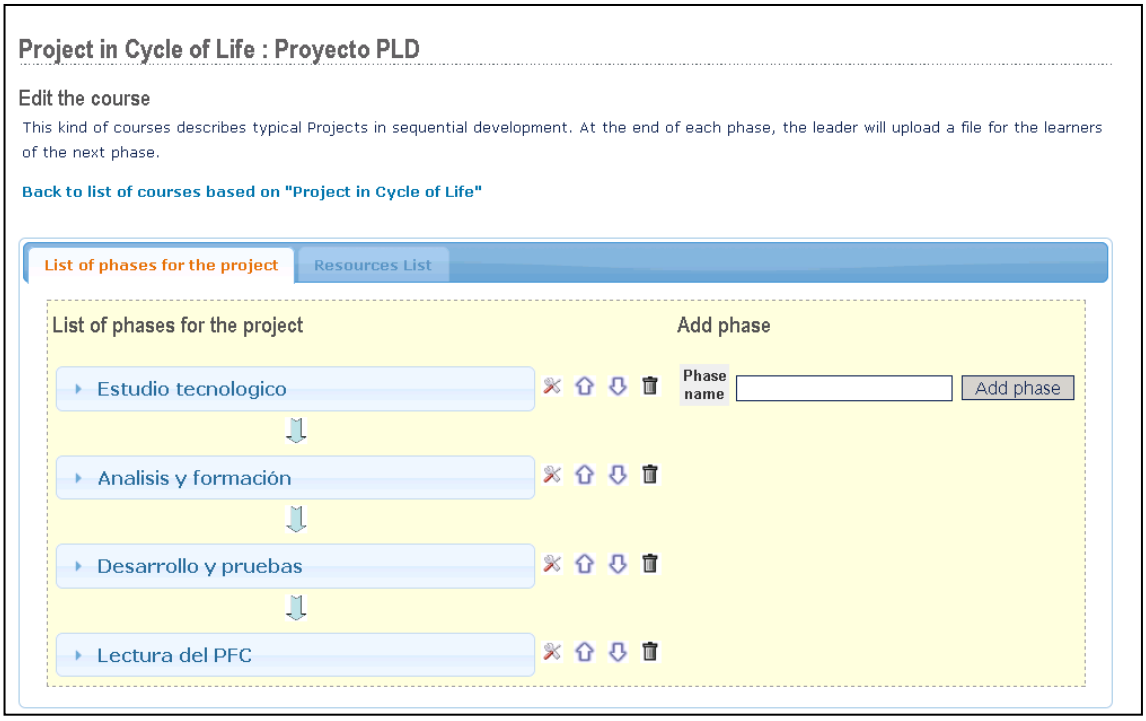


Figura Ap-27. Vista general de la aplicación para la edición de cursos basados en pedagogía orientada a proyectos en ciclo de vida

Haciendo click sobre el nombre de las fases del proyecto, se desplegarán los recursos de cada una de ellas:

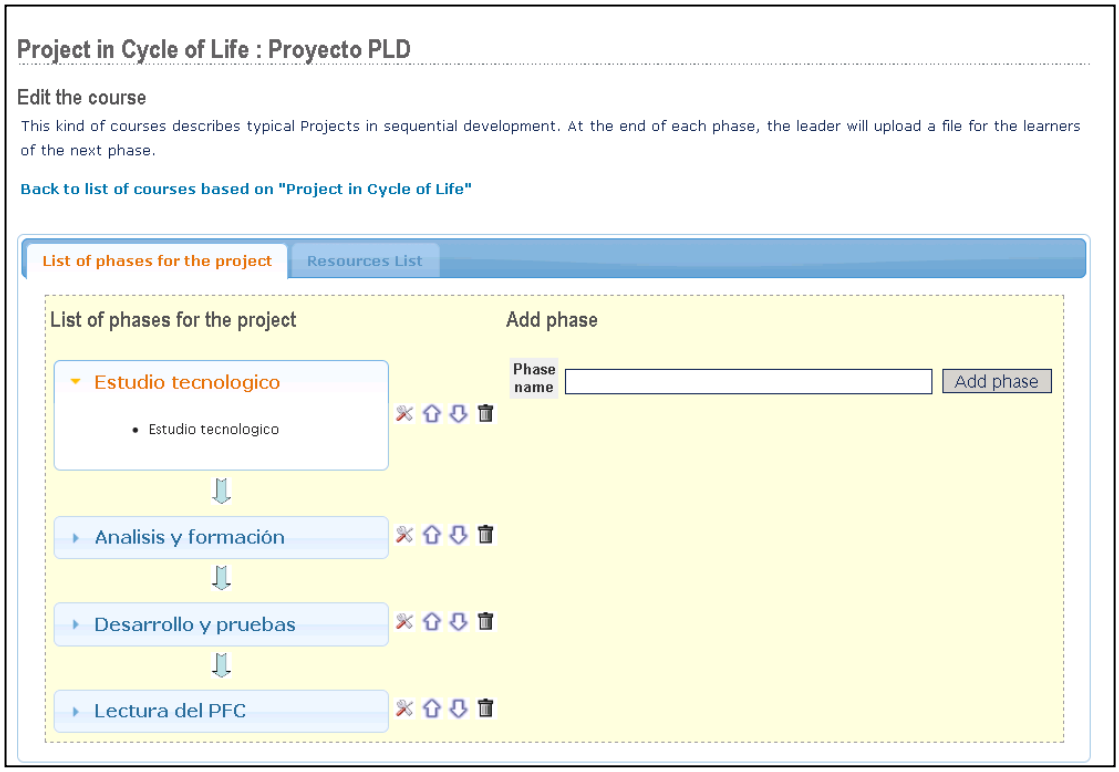


Figura Ap-28. Detalle de la vista para cursos basados en pedagogía orientada a proyectos en ciclo de vida

Para cada una de las fases del proyecto se ofrecen las siguientes opciones:

- Subir > adelanta el orden de ejecución de una fase. Cuanto más arriba, antes se inicia la fase.
- Bajar > atrasa el orden de ejecución de una fase. Cuanto más abajo, más tarde se inicia la fase.
- Editar > activa la pestaña de la derecha para permitir la edición de la fase seleccionada.
- Borrar > elimina la fase y los recursos que tuviera asociados.

Cuando el docente acceda a la opción de edición de una de las fases, la segunda pestaña se activará y mostrará el conjunto de recursos asignados a la fase:

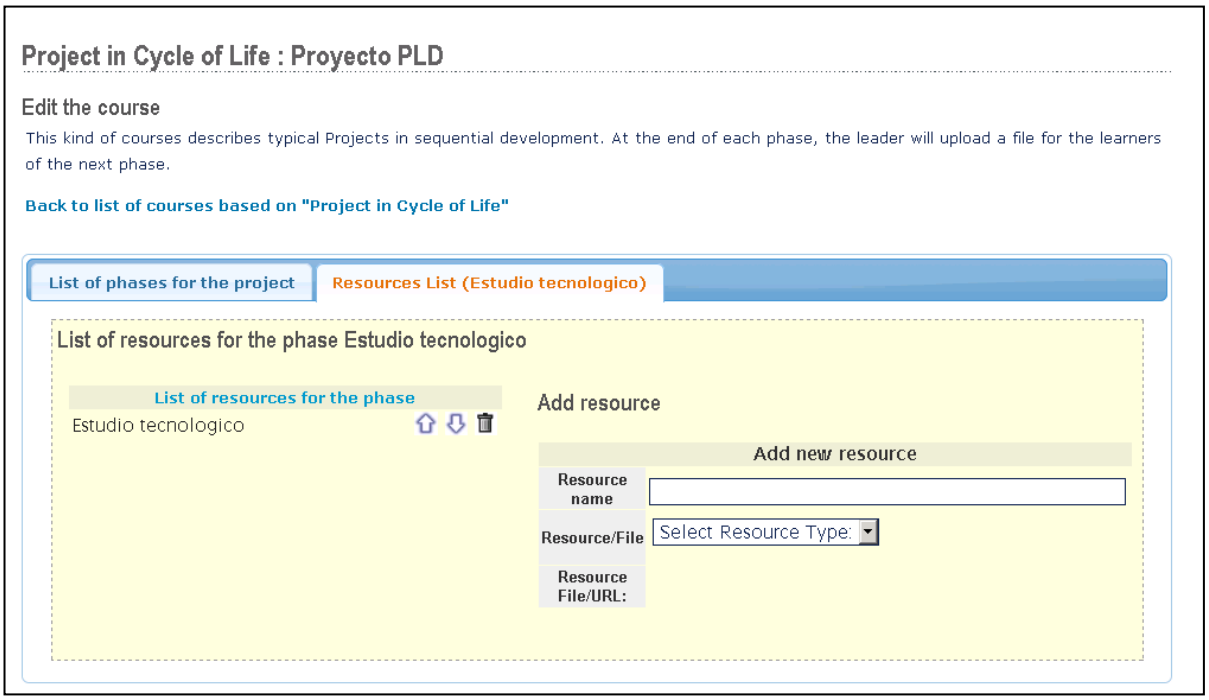


Figura Ap-29. Detalle de la vista para la edición de los recursos de cada fase en la pedagogía orientada a proyectos en ciclo de vida

Desde esta pestaña, el docente usuario de la aplicación podrá añadir nuevos recursos o borrar alguno de los ya existentes. Adicionalmente, podrá establecer el orden de aparición de los mismos.

El contenido del archivo comprimido entregado al docente cuando decida descargarse el curso será similar al siguiente:

- imsmanifest.xml que contiene el código IMS LD con la información del curso.
- ficheros subidos por el profesor
- upload-X.xml que contiene la información necesaria para que los alumnos puedan entregar sus archivos en tiempo de ejecución.
- res-forum-phase-X.html que contiene un sencillo texto de invitación a los diferentes foros en los que se espera que participe el alumno
- res-upload-phase-X(.hml,.xml) implementan la transferencia de información desde el lider de la fase previa
- res-reveive-phase-X(.hml,.xml) implementan la recepción de información por parte del lider de la fase de llegada
- res-chief-view.html permite la monitorización del proyecto por parte del docente

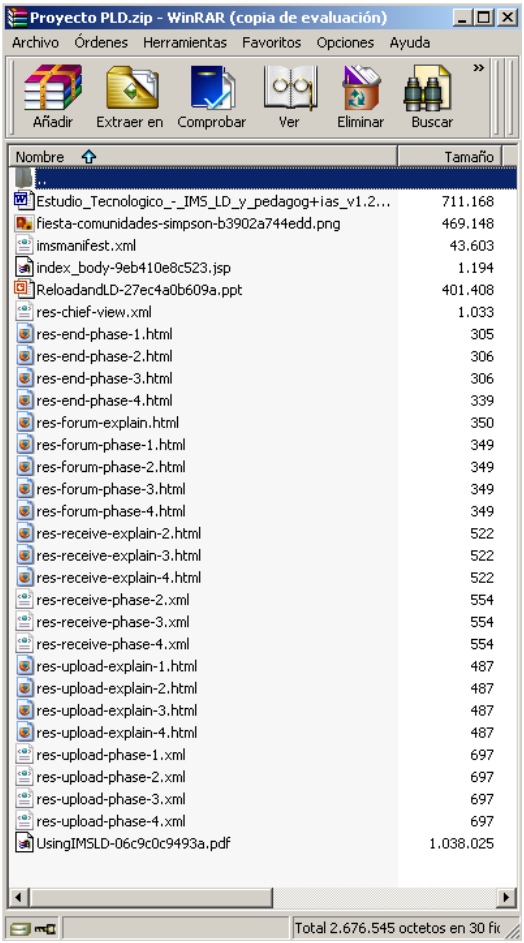


Figura Ap-30. Contenido del fichero comprimido generado por la aplicación para la pedagogía orientada a proyectos en desarrollo secuencial

Apéndice C. MANUAL DEL DESARROLLADOR

C.1. DESARROLLO DE UNA NUEVA PEDAGOGÍA

La arquitectura elegida para el desarrollo de esta "Herramienta de Autor generadora de IMS LD", permite que se pueda cumplir uno de los requisitos previos marcados en el punto 1.2 el requisito de poder se extendida en un futuro para implementar nuevas pedagogías.

En esta aplicación, expansibilidad significa que cualquier desarrollador futuro podría ser capaz de ampliar la utilidad de la herramienta, añadiendo nuevas pedagogías o incluso podría generar código de otras especificaciones distintas a IMS LD.

En la siguiente figura se muestra la arquitectura simplificada de la aplicación, cuando un cliente quiere descargarse el curso una vez editado:

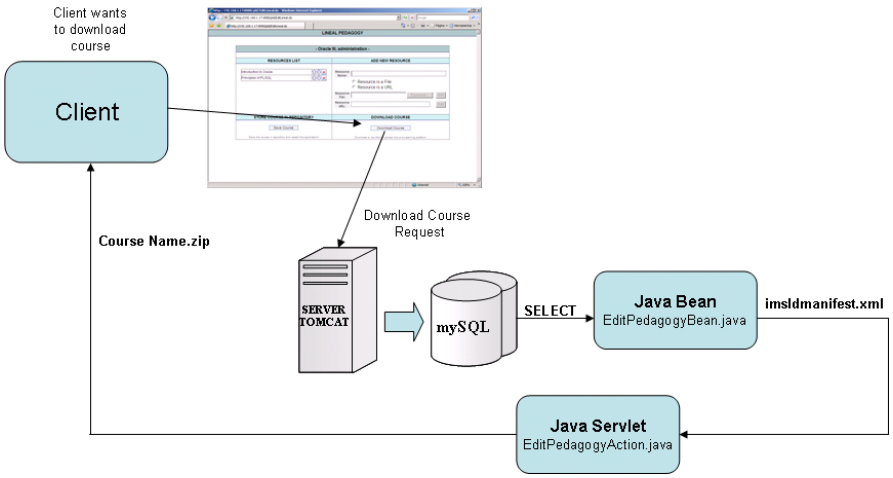


Figura Ap-31. Esquema de cómo se entrega el archivo .zip al cliente.

De la figura anterior, es interesante notar que existen dos Java con cometidos diferentes:

- `EditPedagogyBean.java` se encaga de generar el código IMS LD a partir de los datos almacenados en la base de datos. Concretamente, genera el archivo imsmanifest.xml que contiene toda la lógica de diseño de aprendizaje y lo almacena en la ruta del repositorio de datos asignada para el curso que se esté editando.
- `EditPedagogyAction.java` tiene el propósito fundamental de controlar la interacción con el cliente. Cuando se produce la descarga de un curso por parte del cliente, esta clase también se encargará de generar el archivo comprimido y de entregárselo al cliente.

De la figura anterior, podemos extraer la conclusión de que la lógica de negocio de la Herramienta (la clase EditPedagogyBean.java) está separada de las clases de interacción con el cliente. Por lo que es relativamente sencillo generar nuevas pedagogías. Mostramos a continuación los pasos a seguir para crear e insertar nuevas pedagogías.

Supongamos que queremos añadir una nueva pedagogía a la aplicación. Se trataría de la "Teoría de la atribución" de B. Weiner. Veamos cuáles serían los pasos para añadir esta pedagogía a la Herramienta.

1. *Elección de un nombre para la pedagogía.*

El nombre elegido para la nueva pedagogía debe ser una única palabra que identifique a la nueva pedagogía. Por ejemplo: "lineal", "polya", etc... Para este caso, vamos a elegir la palabra: weiner.

2. *Inserción de la nueva pedagogía en el menú inicial.*

Es necesario añadir un nuevo botón radio en el archivo start.jsp:

```
<div>
        <input type="radio" id="weiner" name="element" value="weiner" />
        <label for="weiner">Attribution Theory (B. Weiner) </label>
</div>
```

Escribiremos la palabra que identifica la pedagogía (weiner para este caso) en los campos requeridos.

3. *Creación de las clases Java necesarias.*

Cada pedagogía requiere tres clases. Para este caso concreto, estas clases Java deben crearse en la carpeta:

```
~ src/pedagogies/weiner/
```

En este caso se deberían crear las siguientes clases:

- EditWeinerForm.java
- EditWeinerAction.java
- EditWeinerBean.java

4. *Selección de una denominación equivalente para la pedagogía.*

Este paso es opcional y sirve para que la aplicación muestre a través de las distintas pantallas el nombre completo de la pedagogía. Para ello es necesario editar el archivo src/utils/Utils.java y añadir lo siguiente:

```
if(s.equalsIgnoreCase("weiner")){
        return "Attribution Theory (B. Weiner)";
}
```

5. *Creación de los ficheros JSP necesarios para la pedagogía.*

En función de la pedagogía que se desea implementar, será necesario un cierto número de archivos JSP para atender la manera en la que el profesor tiene que introducir los parámetros del curso. Lo normal, para simplificar la aplicación de cara al profesor es que el número de JSP sea de uno o dos, aunque son permitidos tantos como sea necesario.

Los archivos JSP deben crearse en la carpeta:

```
~ src/pages/
```

En este caso se deberían crear los siguientes JSP:

- weiner1.jsp
- weiner2.jsp
- ...y así sucesivamente numerados.

6. *Modificar el archivo struts-config.xml para controlar el flujo de la aplicación.*

El fichero `~WebContent/WEB-INF/struts-config.xml` almacena el flujo de la aplicación. Dicho de otro modo, almacena la secuencia de ficheros JSP que deben aparecer en función de las acciones que vaya ejecutando el cliente. Habría que añadir la siguiente etiqueta dentro de `<form-beans>`:

```
<form bean name="EditWeinerForm" type="src.pedagogies.weiner.EditWeinerForm" />
```

y por otro lado, habría que añadir la siguente etiqueta `<action>` dentro de `<action-mappings>`:

```
<action name="EditWeinerForm" path="/EditWeiner" scope="request"
type="src.pedagogies.weiner.EditWeinerAction">
        <forward name="addProblem" path="/pages/weiner2.jsp" />
        <forward name="upProblem" path="/pages/weiner1.jsp" />
        <forward name="downProblem" path="/pages/weiner1.jsp" />
        <forward name="deleteProblem" path="/pages/weiner1.jsp" />
        <forward name="continue" path="/pages/weiner1.jsp" />
        <forward name="editProblem" path="/pages/weiner2.jsp" />
        <forward name="save" path="/pages/start.jsp" />
        <forward name="error" path="/pages/error.jsp" />
</action>
```

Finalmente, en este mismo archivo de configuración `struts-config.xml` hay que añadir las siguientes etiquetas, tanto al action de `CreateCourseForm` como al action de `StartForm`. En este caso sólo se trata de añadir etiquetas `<forward>` dentro de los action:

```
<action name="CreateCourseForm" path="/CreateCourse" scope="request" type="src.CreateCourseAction">
    <forward name="lineal" path="/pages/lineal1.jsp" />
    <forward name="connectionist" path="/pages/connectionist1.jsp" />
    <forward name="polya" path="/pages/polya1.jsp" />
    (...)
    <forward name="weiner" path="/pages/weiner1.jsp" />
</action>
```

```
<action name="StartForm" path="/Start" scope="request" type="src.StartAction">
    <forward name="lineal" path="/pages/lineal1.jsp" />
    <forward name="connectionist" path="/pages/connectionist1.jsp" />
    <forward name="polya" path="/pages/polya1.jsp" />
    (...)
    <forward name="weiner" path="/pages/weiner1.jsp" />
</action>
```

Después de completar estos pasos ya estaría preparada la estructura para desarrollar una nueva pedagogía. Es fundamental dar de alta estos nuevos beans en los ficheros de configuración de Spring como se muestra a continuación para el ejemplo de la pedagogía “Polya”:

applicationContext-dao.xml

```
    <bean id="problemsPedPolyaDAO" class="src.dao.hibernate.ProblemsPedPolyaDAOImpl">
        <property name="sessionFactory" ref="hibernate" />
    </bean>
```

applicationContext-servicios.xml

```
    <bean id="problemsPedPolyaServicio" class="src.servicios.ProblemsPedPolyaServicioImpl">
      <property name="problemsPedPolyaDAO" ><ref bean="problemsPedPolyaDAO"/></property>
    </bean>
```

applicationContext-struts.xml

```
    <bean name="/EditPolya" class="src.pedagogies.polya.EditPolyaAction">
        <property name="problemsPedPolyaSrv" ref="problemsPedPolyaServicio" />
    </bean>
```

C.2. VALIDACIONES DE CAMPOS DE FORMULARIOS

El *framework* Struts propone un potente marco de validaciones tanto en el lado del cliente como en el lado del servidor. Esto supone que con sencillas etiquetas de Struts podemos llegar a generar validaciones en el lado del cliente sin que esto suponga necesariamente tener que desarrollar código Javascript.

Para ello, Struts utiliza archivos de configuración. En el caso de este proyecto, se utilizan los siguientes:

- validation.xml
  - Almacena los formularios que serán validados y el tipo de validación de los diferentes campos. Por ejemplo: campo vacío, campo numérico, etc...
- validation-rules.xml
  - Almacena las reglas que se aplican para la validación. Struts incorpora de manera predefinida las validaciones más comunes, como por ejemplo, comprobar que un campo de un formulario no quede vacío.

Veamos, por ejemplo, la configuración para la validación de un campo de formulario. Por ejemplo para la pedagogía *Connectionist*, durante el proceso de adición de nuevos bloques, no estará permitido dejar el nombre de bloque en blanco.

- validation.xml

```
<form name="EditConnectionistForm">
        <field property="blockName" depends="required">
                <arg0 key="connectionist.edit.blockName" />
        </field>
</form>
```

- validation-rules.xml

```
  <validator name="required"
        classname="org.apache.struts.validator.FieldChecks"
           method="validateRequired"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
                msg="errors.required"/>
```

Una vez realizada la configuración de los XML propios del *framework* Validator de Struts, el siguiente paso será emplearlo en el JSP correspondiente. Por un lado, al comienzo del JSP será necesario introducir la siguiente etiqueta:

```
<html:javascript formName="EditConnectionistForm" />
```

Que será la que incorpore en el HTML generado al cliente el código Javascript necesario para la validación del campo del formulario. Por último, se añade en la propia etiqueta del formulario la llamada al método de validación:

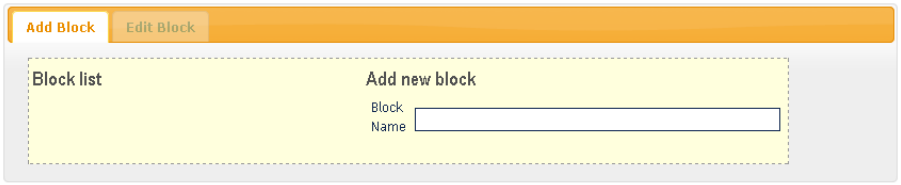
```
<html:form action="EditConnectionist" onsubmit="return validateEditConnectionist(this)">
```

C.3. USO DE JQUERY UI

JQuery UI es un conjunto de librerías y archivos Javascript que permiten desarrollar interfaces de usuario complejas con sencillez. Uno de los principales usos de JQuery a lo largo de esta herramienta de autor es el component JQuery TABS que nos permitirá desplegar elementos de tipo pestaña para ubicar diferentes secciones a la hora de crear una pedagogía. La versión utilizada de JQuery es la 1.7.2. El estilo está almacenado en la siguiente hoja de estilos:

```
/layouts/src/css/ui.theme.css
```

La utilización de JQuery tabs es bastante intuitiva. Para obtener una visualización como la siguiente:



por un lado, en una lista se describen cuáles serán los títulos de los tabs:

```
<ul class="tabs-nav">
     <li><a href="#fragment-addBlock"><span>Add Block</span></a></li>
     <li><a href="#fragment-editBlock" ><span>Edit Block</span></a></li>
</ul>
```

el siguiente paso es agrupar el contenido que debe aparecer dentro de los tabs en una estructura como la siguiente:

```
        <div id="container">
        <div id="fragment-addBlock">[Contenido tab1]</div>
        <div id="fragment-editBlock">[Contenido tab1]</div>
        </div>
```

finalmente, para que los tabs se inicialicen, será necesario introducir la siguiente instrucción de javascript:

```
<script type="text/javascript">
        $(function() {
                $('#container').tabs({ disabled: [1] });
        });
</script>
```

de esta manera, el tab de “Edit Block” aparecería deshabilitado por defecto.

Para majerar qué pestañas deben estar activas y qué pestañas deben estar mostradas/ocultas, se ha desarrollado una clase `TabStatus` que básicamente tiene el siguiente constructor:

```
    public TabStatus(int... flag) {
            for (int aux : flag)
                    addToStatus("" + aux);
            makeDisabled();
            makeSelected();
    }
```

Al constructor se le pasa una lista de enteros (1 o 0) y deben ser cadenas pares. Por ejemplo:

```
1 , 1 , 0 , 1 , 0 , 0
```

Los tres primeros 'bits' (1,1,0) indican que los tabs 0 y 1 deben estar activados.

Los tres ultimos 'bits' (1,0,0) indican que el tab 0 tiene que estar seleccionado.

La clase TabStatus dispone de dos propiedades:

- 'disabled' --> en este ejemplo será el String: "[2]". Indicando que el tab numero 2 tiene que estar deshabilitado.
- 'selected' --> en este caso será el String: "0".

La forma de integrarlo con jQuery es muy sencilla ya que en el JSP que queramos usar sólo tenemos que hacer lo siguiente:

```
<c:choose>
    <!—CASO POR DEFECTO QUE SE CONTEMPLA A PARTE -->
    <c:when test="${tabStatus==null}">
            <script type="text/javascript">
                $(function() {
                    $('#container').tabs();
                    $('#container').tabs('option', 'disabled', [1,2]);
                    $('#container').tabs('select', 0);
                });
             </script>
    </c:when>
    <!—EL CASO GENERAL-->
    <c:otherwise>
            <script type="text/javascript">
                $(function() {
                    $('#container').tabs();
                    $('#container').tabs('option', 'disabled', ${tabStatus.disabled});
                    $('#container').tabs('select', ${tabStatus.selected});
                });
             </script>
    </c:otherwise>
</c:choose>
```

Desde un action cualquiera podemos hacer una llamada como la siguiente:

```
// Editing actions
if (action.equals("editPart")) {
    PartsPedProsim parte =
partsPedProsimservicio.getPart(request.getParameter("identifier"));
    session.setAttribute("partToEdit", parte);
    action = "continue";
    session.setAttribute("tabStatus", new TabStatus(1, 1, 0, 0, 1, 0));
}
```

De esta manera no tenemos que estar preocupándonos de qué tab es el que se muestra en cada momento ya que de manera sencilla podemos manejar tanto el estado "enabled" o "disabled" así como el "selected".

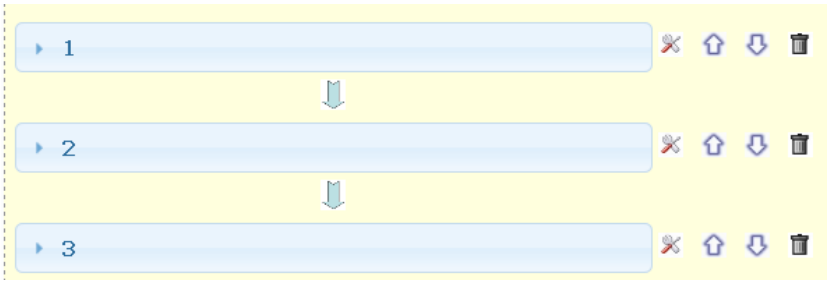
JQuery UI también ha sido utilizado en el desarrollo de la pedagogía orientada a proyectos en desarrollo simultáneo para agrupar los bloques que representan las fases del proyecto. Se ha utilizado un componente de JQuery llamado "acordion". Como su nombre indica se trata de secciones de la página que permanecen ocultas esperando un evento por parte del usuario para ser mostrados.

El acordion tiene la siguiente estructura en la definición e inicialización:

```
<c:if test="${fn:length(phases)>0}">
    <script type="text/javascript">
            $(function() {
            <c:forEach items="${phases}" var="phase" varStatus="status">

                    $('#accordion${status.index}').accordion({
collapsible:true ,
active: false ,
autoHeight: false });
            </c:forEach>
            });
        </script>
</c:if>
```

Por otro lado, se declaran los elementos *acordion* dentro de una estructura de tabla para poder tener los iconos habituales:




```
<c:if test="${fn:length(phases)>0}">
    <table width="100%">
    <c:forEach items="${phases}" var="phase" varStatus="status">
            <tr>
                    <td width="80%">
                            <div id="accordion${status.index}">
                                    <h3><a href="#">${phase.name}</a></h3>
                                    <div>
                                    <ul style="margin-top:0px;font-size:10px;">
                                            <c:forEach items="${phase.resources}"
var="resource" varStatus="statusIn">
                                                    <li>${resource.name}</li>

                                            </c:forEach>
                                    </ul>
                                    </div>
                            </div>
                    </td>
                    <td>[imagen1]</td>
                    <td>[imagen2, etc...]</td>
            </tr>
            <c:if test="${not status.last}">
            <tr>
                    <td>[imagen de conexión entre fases]</td>
                    <td colspan="4"> </td>
            </tr>
            </c:if>
    </c:forEach>
    </table>
</c:if>
```

Apéndice D. EJEMPLOS DE FICHEROS XML IMS-LD GENERADOS CON LA HERRAMIENTA DE ACUERDO A LAS DIFERENTES PEDAGOGÍAS

A.1. IMSMANIFEST.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" identifier="COURSE-PBL-PEDAGOGY-
Redes_y_sistemas">
<!--Redes y sistemas. File automatically generated. Created: 04 abr 2011 01:09:51 +0200-->
<!--PFC: Desarrollo de una Herramienta de Autor generadora de IMS LD para diferentes pedagogias--
>
<!--Alumn: Francisco J. Murcia Sanchez / Tutor: Pedro Munoz Merino-->
<!--Universidad Carlos III de Madrid (Departamento de Ingenieria Telematica) SPAIN-->
<organizations>
<imsld:learning-design xmlns:imsld="http://www.imsglobal.org/xsd/imsld_v1p0"
identifier="Redes_y_sistemas" level="B" uri="http://www.pld-project.net/uri/LD-2f419e4181fe">
<imsld:title>Redes y sistemas</imsld:title>
<imsld:components>
<imsld:roles>
  <imsld:learner identifier="Learner">
    <imsld:title>Learner</imsld:title>
  </imsld:learner>
  <imsld:staff identifier="Staff">
    <imsld:title>Teacher</imsld:title>
  </imsld:staff>
</imsld:roles>
<imsld:properties>
  <imsld:loc-property identifier="Statement-1">
    <imsld:title>Problem 1 Statement</imsld:title>
    <imsld:datatype datatype="string" />
  </imsld:loc-property>
  <imsld:loc-property identifier="Statement-1_fin">
    <imsld:title>Problem 1 Statement</imsld:title>
    <imsld:datatype datatype="string" />
    <imsld:initial-value>NO</imsld:initial-value>
    <imsld:restriction restriction-type="enumeration">YES</imsld:restriction>
    <imsld:restriction restriction-type="enumeration">NO</imsld:restriction>
  </imsld:loc-property>
  <imsld:locpers-property identifier="Comment-1">
    <imsld:title>Your comments</imsld:title>
    <imsld:datatype datatype="string" />
  </imsld:locpers-property>
  <imsld:locpers-property identifier="File-1">
    <imsld:title>Your assessment</imsld:title>
    <imsld:datatype datatype="file" />
  </imsld:locpers-property>
  <imsld:property-group identifier="Assessment-1">
    <imsld:title>Grupo de propiedades</imsld:title>
    <imsld:property-ref ref="Comment-1" />
    <imsld:property-ref ref="File-1" />
  </imsld:property-group>
</imsld:properties>
<imsld:activities>
  <imsld:learning-activity identifier="LA-s1-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="true">
    <imsld:title>Read statement: Situacion actual</imsld:title>
    <imsld:activity-description>
      <imsld:item identifier="I-s1-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" identifierref="R-
ad567ec2-1bd2-4195-9d9c-583603bcb8c8" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>
  <imsld:learning-activity identifier="LA-s2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="true">
    <imsld:title>What do we know?</imsld:title>
    <imsld:environment-ref ref="E-Forum-S2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:activity-description>
```

```
      <imsld:item identifier="I-s2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" identifierref="RES-
STEP2-FORUM-P1" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>

   <imsld:learning-activity identifier="LA-s3-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="true">
    <imsld:title>Develop statement</imsld:title>
    <imsld:activity-description>
      <imsld:item identifier="I-s3-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" identifierref="R-s3-
1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>

    <imsld:learning-activity identifier="LA-s3-teacher-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="true">
    <imsld:title>Develop statement</imsld:title>
    <imsld:activity-description>
      <imsld:item identifier="I-s3-teacher-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
identifierref="R-s3-teacher-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>

  <imsld:learning-activity identifier="LA-s4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="false">
    <imsld:title>What do we need to know?</imsld:title>
    <imsld:environment-ref ref="E-Forum-S4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:activity-description>
      <imsld:item identifier="I-s4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" identifierref="RES-
STEP4-FORUM-P1" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>
  <imsld:learning-activity identifier="LA-s5-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="false">
    <imsld:title>Self study. P1</imsld:title>
    <imsld:environment-ref ref="E-Study-S5-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:activity-description>
      <imsld:item identifier="I-s5-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" identifierref="RES-
STEP5-P1" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>
  <imsld:learning-activity identifier="LA-s6-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
isvisible="false">
    <imsld:title>Upload solution. P1</imsld:title>
    <imsld:environment-ref ref="E-Assessment-Problem1" />
    <imsld:activity-description>
      <imsld:item identifier="I-s6-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" identifierref="RES-
STEP6-P1" />
    </imsld:activity-description>
    <imsld:complete-activity>
      <imsld:user-choice />
    </imsld:complete-activity>
  </imsld:learning-activity>

  <imsld:activity-structure identifier="AS-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" structure-
type="sequence">
    <imsld:title>Bajo rendimiento de la red</imsld:title>
    <imsld:learning-activity-ref ref="LA-s1-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:learning-activity-ref ref="LA-s2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:learning-activity-ref ref="LA-s3-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />

  </imsld:activity-structure>
```

```
  <imsld:activity-structure identifier="AS-2" structure-type="sequence">
    <imsld:learning-activity-ref ref="LA-s4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:learning-activity-ref ref="LA-s5-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:learning-activity-ref ref="LA-s6-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
  </imsld:activity-structure>

  <imsld:activity-structure identifier="AS-TEACHER-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
structure-type="selection">
    <imsld:title>Bajo rendimiento de la red. Teacher view</imsld:title>
    <imsld:learning-activity-ref ref="LA-s2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:learning-activity-ref ref="LA-s3-teacher-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    <imsld:learning-activity-ref ref="LA-s4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
  </imsld:activity-structure>

  <imsld:support-activity identifier="SA-monitor_progress" isvisible="true">
    <imsld:title>Monitor Progress</imsld:title>
    <imsld:role-ref ref="Staff" />
    <imsld:environment-ref ref="E-monitor_progress" />
    <imsld:activity-description>
      <imsld:item identifier="I-monitor_progress" identifierref="R-monitor_progress" />
    </imsld:activity-description>
  </imsld:support-activity>

  <imsld:activity-structure identifier="AS-MONITOR" structure-type="selection">
    <imsld:title>Monitor Progress</imsld:title>
    <imsld:support-activity-ref ref="SA-monitor_progress" />
  </imsld:activity-structure>
  <imsld:activity-structure structure-type="sequence" identifier="AS-COMPLETE-PBL">
    <imsld:title>Problem based learning for engineers</imsld:title>
    <imsld:activity-structure-ref ref="AS-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
  </imsld:activity-structure>
  <imsld:activity-structure structure-type="sequence" identifier="AS-COMPLETE-PBL-2">
    <imsld:title>Problem based learning for engineers</imsld:title>
    <imsld:activity-structure-ref ref="AS-2" />
  </imsld:activity-structure>
  <imsld:activity-structure structure-type="selection" identifier="AS-COMPLETE-PBL-TEACHER">
    <imsld:title>Problem based learning for engineers. Teacher view.</imsld:title>
    <imsld:activity-structure-ref ref="AS-TEACHER-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
  </imsld:activity-structure>
</imsld:activities>
<imsld:environments>
  <imsld:environment identifier="E-Forum-S2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16">
    <imsld:title>Forum. List what is known</imsld:title>
    <imsld:service identifier="Serv-1a6faa28-1f93-4ceb-bc33-cd24b44eed16">
      <imsld:conference conference-type="asynchronous">
        <imsld:title>Discuss about: What do we know?</imsld:title>
        <imsld:participant role-ref="Learner" />
        <imsld:moderator role-ref="Staff" />
        <imsld:item identifier="ID-CONF-s2-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
identifierref="RES-STEP2-FORUM-P1" />
      </imsld:conference>
    </imsld:service>
  </imsld:environment>
  <imsld:environment identifier="E-Forum-S4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16">
    <imsld:title>Forum. What do we need to know?</imsld:title>
    <imsld:service identifier="Serv-4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16">
      <imsld:conference conference-type="asynchronous">
        <imsld:title>Discuss about: What do we need to know?</imsld:title>
        <imsld:participant role-ref="Learner" />
        <imsld:moderator role-ref="Staff" />
        <imsld:item identifier="ID-CONF-s4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16"
identifierref="RES-STEP4-FORUM-P1" />
      </imsld:conference>
    </imsld:service>
  </imsld:environment>
  <imsld:environment identifier="E-Study-S5-1a6faa28-1f93-4ceb-bc33-cd24b44eed16">
    <imsld:title>Contents for self study</imsld:title>
    <imsld:learning-object identifier="Sup_P1_1_Referencia_de_sistemas_operativos_en_red">
      <imsld:item identifierref="R-2e35e461-3297-4486-9a39-aedea70c4c1d" identifier="I-2e35e461-
3297-4486-9a39-aedea70c4c1d" />
    </imsld:learning-object>
  </imsld:environment>
  <imsld:environment identifier="E-Assessment-Problem1">
    <imsld:title>Upload your assessment here</imsld:title>
    <imsld:learning-object identifier="Assessment-problem-1">
      <imsld:item identifierref="R-Assessment-problem1" identifier="I-Assessment-problem1" />
```

```
    </imsld:learning-object>
  </imsld:environment>
  <imsld:environment identifier="E-monitor_progress">
    <imsld:title>Monitor</imsld:title>
    <imsld:service identifier="Services-1cdd5fe6-87cf-4982-9fc8-45525ba3a48e">
      <imsld:monitor>
        <imsld:role-ref ref="Learner" />
        <imsld:title>Monitor</imsld:title>
        <imsld:item identifier="ID-Service-Monitor" identifierref="R-monitor_progress" />
      </imsld:monitor>
    </imsld:service>
  </imsld:environment>
</imsld:environments>
</imsld:components>
<imsld:method>
<imsld:play>
  <imsld:title>Course: Redes y sistemas</imsld:title>
  <imsld:act>
    <imsld:title>Curso de Redes y sistemas</imsld:title>
    <imsld:role-part>
      <imsld:title>Role part learner</imsld:title>
      <imsld:role-ref ref="Learner" />
      <imsld:activity-structure-ref ref="AS-COMPLETE-PBL" />
    </imsld:role-part>
    <imsld:role-part>
      <imsld:title>Role Part Teacher</imsld:title>
      <imsld:role-ref ref="Staff" />
      <imsld:activity-structure-ref ref="AS-COMPLETE-PBL-TEACHER" />
    </imsld:role-part>
    <imsld:role-part>
      <imsld:title>Monitor</imsld:title>
      <imsld:role-ref ref="Staff" />
      <imsld:activity-structure-ref ref="AS-MONITOR" />
    </imsld:role-part>
    <imsld:complete-act>
       <imsld:when-condition-true>
              <imsld:role-ref ref="Staff" />
              <imsld:property-ref ref="Statement-1_fin" />
              <imsld:property-value>YES</imsld:property-value>
       </imsld:when-condition-true>
   </imsld:complete-act>
  </imsld:act>
  <imsld:act>
    <imsld:title>Curso de Redes y sistemas</imsld:title>
    <imsld:role-part>
      <imsld:title>Role part learner</imsld:title>
      <imsld:role-ref ref="Learner" />
      <imsld:activity-structure-ref ref="AS-COMPLETE-PBL-2" />
    </imsld:role-part>
    <imsld:role-part>
      <imsld:title>Role Part Teacher</imsld:title>
      <imsld:role-ref ref="Staff" />
      <imsld:activity-structure-ref ref="AS-COMPLETE-PBL-TEACHER" />
    </imsld:role-part>
    <imsld:role-part>
      <imsld:title>Monitor</imsld:title>
      <imsld:role-ref ref="Staff" />
      <imsld:activity-structure-ref ref="AS-MONITOR" />
    </imsld:role-part>
  </imsld:act>
</imsld:play>
<imsld:conditions>
  <imsld:if>
    <imsld:is>
      <imsld:property-ref ref="Statement-1_fin" />
      <imsld:property-value>YES</imsld:property-value>
    </imsld:is>
  </imsld:if>
  <imsld:then>
    <imsld:show>
      <imsld:learning-activity-ref ref="LA-s4-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    </imsld:show>
  </imsld:then>
  <imsld:if>
    <imsld:is>
      <imsld:property-ref ref="Statement-1_fin" />
      <imsld:property-value>YES</imsld:property-value>
```

```
    </imsld:is>
  </imsld:if>
  <imsld:then>
    <imsld:show>
      <imsld:learning-activity-ref ref="LA-s5-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    </imsld:show>
  </imsld:then>
  <imsld:if>
    <imsld:is>
      <imsld:property-ref ref="Statement-1_fin" />
      <imsld:property-value>YES</imsld:property-value>
    </imsld:is>
  </imsld:if>
  <imsld:then>
    <imsld:show>
      <imsld:learning-activity-ref ref="LA-s6-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" />
    </imsld:show>
  </imsld:then>
  <imsld:if>
    <imsld:is>
      <imsld:property-ref ref="Statement-1_fin" />
      <imsld:property-value>YES</imsld:property-value>
    </imsld:is>
  </imsld:if>
  <imsld:then>
    <imsld:hide>
      <imsld:class class="set_statement_problem1" />
    </imsld:hide>
  </imsld:then>
</imsld:conditions>
</imsld:method>
</imsld:learning-design>
</organizations>
<resources>
<resource identifier="RES-STEP2-FORUM-P1" type="forum" href="res-step2-forum1.html">
<file href="res-step2-forum1.html" />
</resource>
<resource identifier="R-s3-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" type="imsldcontent" href="step3-
P1.xml">
<file href="step3-P1.xml" />
</resource>
<resource identifier="R-s3-teacher-1a6faa28-1f93-4ceb-bc33-cd24b44eed16" type="imsldcontent"
href="step3-P1_teacher.xml">
<file href="step3-P1_teacher.xml" />
</resource>
<resource identifier="RES-STEP4-FORUM-P1" type="forum" href="res-step4-forum1.html">
<file href="res-step4-forum1.html" />
</resource>
<resource identifier="RES-STEP5-P1" type="webcontent" href="res-step5-p1.html">
<file href="res-step5-p1.html" />
</resource>
<resource identifier="R-Assessment-problem1" type="imsldcontent" href="upload-1.xml">
<file href="upload-1.xml" />
</resource>
<resource identifier="RES-STEP6-P1" type="webcontent" href="res-step6-P1.html">
<file href="res-step6-P1.html" />
</resource>
<resource identifier="R-monitor_progress" type="imsldcontent" href="monitor.xml">
<file href="monitor.xml" />
</resource>
<resource identifier="R-ad567ec2-1bd2-4195-9d9c-583603bcb8c8" type="webcontent" href="enunciado-
222b3b409bdd.html">
<file href="enunciado-222b3b409bdd.html" />
</resource>
<resource identifier="R-2e35e461-3297-4486-9a39-aedea70c4c1d" type="webcontent"
href="http://es.wikipedia.org/wiki/Sistema_operativo_de_red" />
<resource identifier="RES-STEP2" type="webcontent" href="res-step2.html">
<file href="res-step2.html" />
</resource>
<resource identifier="RES-STEP4" type="webcontent" href="res-step4.html">
<file href="res-step4.html" />
</resource>
</resources>
</manifest>
```

A.2. MONITOR.XML

```
<?xml version="1.0"?>
	<!--
		Redes y sistemas, file automatically generated.
	-->
<html xmlns:ld="http://www.imsglobal.org/xsd/imsld_v1p0" xmlns="http://www.w3.org/1999/xhtml">
	<head>
		<title />
	</head>
	<body>
		<p>
			<font color="#000000" size="3">Comments and assessment for
				problem 1: Bajo rendimiento de la red</font>
		</p>
		<p>
			<ld:view-property-group ref="Assessment-1"
				property-of="supported-person" view="value" />
		</p>
		<hr></hr>
	</body>
</html>
```

A.3. UPLOAD.XML

```
<?xml version="1.0"?>
	<!--
		Redes y sistemas, file automatically generated.
	-->
<html xmlns:ld="http://www.imsglobal.org/xsd/imsld_v1p0" xmlns="http://www.w3.org/1999/xhtml">
	<head>
		<title />
	</head>
	<body>
		<p>
			<font color="#000000" size="3">Please, upload your file and
				comments: </font>
		</p>
		<p>
			<ld:set-property-group ref="Assessment-1"
				property-of="self" view="title-value" />
		</p>
		<hr></hr>
		<p>
			<font color="#000000" size="3">These are your comments and your
				assesment</font>
		</p>
		<p>
			<ld:view-property-group ref="Assessment-1"
				property-of="self" view="title-value" />
		</p>
	</body>
</html>
```