APIS: Una Herramienta de Soporte al Aprendizaje Colaborativo de la Ingeniería del Software

Juan Carlos Yelmo
Dpto. de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Av. Complutense, 30, 28040 Madrid
+34 91 336 6830
jcyelmo@dit.upm.es

Juan Fernández-Corugedo
Dpto. de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Av. Complutense, 30, 28040 Madrid
+34 91 336 6830
jfcorugedo@yahoo.es

RESUMEN

En este artículo se describe la experiencia de innovación educativa en torno al aprendizaje de la materia Ingeniería del Software en la especialidad de Telemática de la titulación de Ingeniero de Telecomunicación en la UPM. En particular, se describe un proyecto de adaptación de la asignatura Laboratorio de Ingeniería del Software a los postulados docentes y organizativos del EEES. En este contexto, se ha desarrollado APIS, una aplicación Web para soporte al aprendizaje colaborativo de la Ingeniería del Software.

Palabras clave

Ingeniería del Software, Ingeniería Telemática, Aprendizaje Colaborativo, Espacio Europeo de Educación Superior.

1. INTRODUCTION

En este artículo se describe la experiencia de innovación educativa en torno al aprendizaje de la materia Ingeniería del Software en la especialidad de Telemática de la titulación de Ingeniero de Telecomunicación, dentro del plan de estudios actual de la UPM (Plan 94), plan que en breve dará paso a la nueva estructura de Grado y Máster propuesta por el plan de convergencia europea en materia de educación. En particular, se describe el contexto de evolución de las asignaturas relacionadas para su adaptación a los postulados docentes y organizativos del Espacio Europeo de Educación Superior, el entorno de laboratorio y de herramientas de soporte al aprendizaje colaborativo de la Ingeniería del Software y, más en concreto, entraremos en detalles en la descripción de la herramienta APIS (Administración de Proyectos en Ingeniería del Software), herramienta Web desarrollada en el Departamento del Ingeniería de Sistemas Telemáticos para soportar la gestión de procesos de desarrollo de software basados en el modelo iterativo propuesto por el denominado Modelo de Proceso Unificado [1].

Esta experiencia de innovación educativa se ha llevado a cabo principalmente en relación con la asignatura *Laboratorio de Ingeniería del Software*. Se trata de una asignatura de quinto curso de la especialidad de Telemática que se imparte desde el curso 98-99 en el contexto de implantación del Plan 94 de la titulación. La asignatura sigue desde el comienzo (anterior a la declaración de Bolonia en junio de 1999) un enfoque de metodología docente centrada en el aprendizaje y el trabajo colaborativo de los alumnos con seguimiento y tutoría por parte de los profesores de la asignatura y evaluación continua. Los alumnos se agrupan en equipos de proyecto, donde cada miembro del equipo tiene asignado un rol y una responsabilidad compartida en la consecución de objetivos globales. A cada equipo se le asigna un proyecto de desarrollo de un sistema software, en general una aplicación distribuida o un servicio telemático, y realizan el proyecto a lo largo del semestre con la tutoría de un profesor supervisor y el soporte de un conjunto de herramientas de ingeniería del software y una plataforma web para publicación de contenidos docentes, gestión de proyecto y trabajo colaborativo distribuido. La evaluación continua se realiza en base a los documentos, demostraciones, presentaciones y resultados entregados por los alumnos a lo largo del semestre.

La asignatura está sometida desde su implantación a un proceso anual de revisión y mejora de sus métodos docentes, esquema organizativo, herramientas y plataforma telemática de soporte. Este proceso de mejora continua se intensificó desde el curso 2005-06, año en que la asignatura ha sido un proyecto piloto de implantación de nuevos métodos docentes y evaluadores vinculados al sistema de créditos europeo (ECTS) en el programa de implantación del EEES y Calidad de la Enseñanza de la UPM. En este contexto, se revisaron sus objetivos, contenidos y estructura organizativa y se rediseñó el entorno de herramientas de soporte a la docencia. En particular, se actualizó una primera versión de APIS, basada en tecnologías Web y Java EE, que actuaba como *front-end* de una plataforma de trabajo colaborativo (BSCW). Esta herramienta se ha utilizado con éxito a lo largo de varios cursos en la impartición del laboratorio y en el curso 2009-10 ha sido rediseñada completamente a la luz de la experiencia de uso con objeto de mejorar su funcionalidad y prestaciones, actualizar sus tecnologías de implementación y entorno de ejecución y hacerla autocontenida e independiente de BSCW, de forma que APIS pueda funcionar con otras plataformas de trabajo colaborativo o de gestión de aprendizaje, en particular Moodle.

En el resto del artículo se proporciona información sobre el contexto docente de la formación en ingeniería del software dentro de la especialidad de Ingeniería Telemática para, a continuación describir en detalle la organización y metodología docente del Laboratorio de Ingeniería del software y el propósito y resultados de este proyecto de innovación educativa, que se

concreta en la herramienta APIS en el contexto del entorno de herramientas de soporte docente del Laboratorio.

2. LA INGENIERÍA DEL SOFTWARE EN INGENIERÍA TELEMÁTICA

El plan de estudios 94 de la ETSIT-UPM contempla la elección de especialidad en el segundo semestre del cuarto curso, momento en que los alumnos pueden optar por una de las tres especialidades de la titulación: Electrónica, Telemática y Comunicaciones. En el caso de la especialidad de Telemática, las asignaturas específicas son *Software de Comunicaciones* e *Ingeniería del Software*. A partir de quinto curso los alumnos cursan el *Laboratorio de Software de Comunicaciones* como obligatoria de especialidad y pueden elegir entre un conjunto de asignaturas optativas agrupadas en dos itinerarios (intensificaciones): *Redes y Servicios de Comunicaciones* y *Sistemas Informáticos*. En función de la definición parcialmente abierta de las intensificaciones y la oferta de optativas comunes, en la práctica ambas intensificaciones se entremezclan en el itinerario académico elegido por los alumnos de la especialidad[1]. En particular, la intensificación de *Sistemas Informáticos* tiene como laboratorio obligatorio el *Laboratorio de Ingeniería del Software*, que también puede cursarse como optativa en la intensificación de *Redes y Servicios de Comunicaciones*. La siguiente figura resumen de forma gráfica la estructura de la especialidad de telemática en el Plan 94.

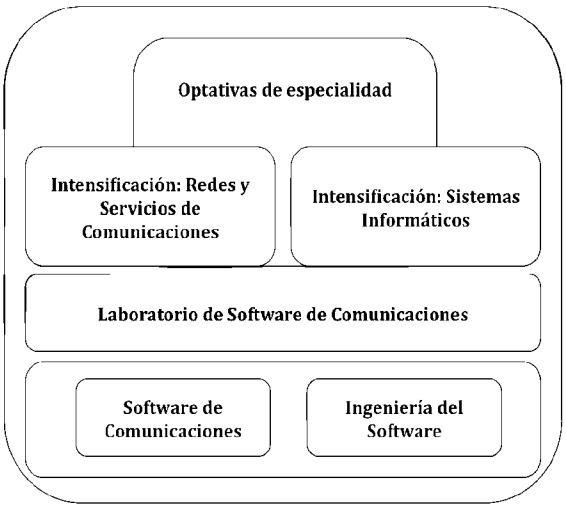

Figura 1. La especialidad de Telemática en el Plan 94

En resumen, el grueso de la formación en ingeniería del software, dentro de la especialidad de telemática, se lleva a cabo en las asignaturas *Ingeniería del Software* y *Laboratorio de Ingeniería del Software*.

La asignatura de *Ingeniería del Software* es una típica asignatura teórica con un programa introductorio, amplio y generalista (es decir no aplicado en exclusiva a sistemas y servicios de telecomunicación) sobre la disciplina. El propósito es dotar a los alumnos de los conocimientos y capacidades necesarios para entender y abordar no sólo las facetas técnicas del desarrollo industrial de sistemas software, sino sus aspectos organizativos, económicos y sociales. La asignatura se estructura en tres grandes bloques temáticos:

- *El proceso de desarrollo de software* [2]. Descripción general de los modelos de ciclo de vida y los diferentes procesos, técnicas y herramientas involucrados en el de desarrollo de software, desde la captura de requisitos a la garantía de calidad de sistemas software. Estos procesos vienen a coincidir básicamente con los procesos del ciclo de vida definidos en el estándar ISO 12207 [3] y con las áreas de conocimiento definidas en el SWEBOK [4].
- *Técnicas de Orientación a Objetos* [1]. En este bloque se describe una metodología general de análisis y diseño de sistemas software en base al Modelo de Proceso Unificado y al enfoque orientado a objetos, con UML como lenguaje de modelado.
- *Gestión de proyectos* [5]. En este bloque se describen conceptos y técnicas de gestión de proyectos de software (estimación, planificación, gestión de riesgos, etc.) junto con aspectos económicos y humanos relacionados con la gestión de equipos de trabajo en proyectos de desarrollo.

El programa de la asignatura se completa con conferencias invitadas en las que profesionales con responsabilidad en materia de ingeniería del software en empresas del sector tratan temas diversos en relación con la práctica profesional o aspectos técnicos avanzados. A modo de ejemplo, algunos de los temas tratados en los últimos cursos han sido: Externalización de actividades de verificación y validación, el estándar ITIL para gestión de servicios TI, arquitecturas de núcleos de ejecución de servicios bancarios, gestión de la innovación y protección de la propiedad industrial e intelectual.

Esta asignatura ha evolucionado en los últimos años hacia un modelo de aprendizaje basado en la evaluación continua y la combinación de clases magistrales con prácticas guiadas y trabajo en grupo.

En cuanto al *Laboratorio de Ingeniería del Software*, su objetivo básico es reforzar el aprendizaje de los conceptos y técnicas de ingeniería del software que los alumnos deben haber adquirido en la asignatura *Ingeniería del Software*, complementando los conocimientos teóricos con habilidades prácticas necesarias para poder utilizar técnicas, metodologías, estándares, notaciones y herramientas de soporte de amplio uso y aceptación en el desarrollo industrial de aplicaciones software.

En el caso del Laboratorio, se concreta el campo de aplicación de la ingeniería del software en los sistemas y servicios de telecomunicación: servicios de Internet móvil, servicios en convergencia, aplicaciones de *cloud computing*, servicios telemáticos para el ciudadano, etc.

La siguiente sección describe en detalle el enfoque organizativo y docente de esta asignatura.

[1] http://www.dit.upm.es/asignaturas-de-grado.html

3. EL LABORATORIO DE INGENIERÍA DEL SOFTWARE

El *Laboratorio de Ingeniería del Software* es una asignatura eminentemente práctica que desarrolla la mayor parte de su programa en forma de sesiones de laboratorio y trabajo colaborativo presencial y remoto libremente organizado por los grupos de prácticas.. Para ello, se propone a los alumnos una serie de casos de estudio de desarrollo software. En general, se proponen ejemplos de aplicaciones distribuidas y servicios telemáticos en convergencia que resulten realistas y motivadores. No obstante, la asignatura pone énfasis en la metodología de desarrollo en si misma y no en el producto software o servicio a desarrollar, limitando el esfuerzo de implementación mediante integración de componentes preexistentes, tanto de desarrollo propio como disponibles en forma de código fuente abierto, y mediante el uso de emuladores de servicios auxiliares (centros de servicios MMS o SMS, servicios de posicionamiento, etc.). En cualquier caso, las prácticas están diseñadas para ilustrar los siguientes aspectos:

- Gestión y planificación de proyectos de desarrollo de software
- Coordinación de equipos de desarrollo y división del trabajo
- Gestión de configuración en proyectos de desarrollo de software
- Uso de estándares de modelado y documentación
- Diseño de arquitectura de aplicaciones y servicios telemáticos
- Análisis y diseño orientado a objetos de una aplicación software
- Uso de herramientas de soporte en las diferentes fases o disciplinas del proceso de desarrollo.

Los casos de estudio han de realizarse siguiendo una planificación, un esquema organizativo y un conjunto de técnicas y herramientas similares a los utilizados en proyectos de desarrollo de software reales en la práctica profesional. Para la realización de los casos de estudio los alumnos constituyen equipos de proyecto compuestos por cuatro o cinco miembros. Los equipos de trabajo se organizan con criterios de reparto de responsabilidades y división del trabajo, por lo que cada miembro del equipo ha de jugar un rol principal. El nombre y funciones específicas de cada rol dependerán del modelo de proceso adoptado, pero en general estarán muy próximos a los siguientes

- **Responsable de Proyecto**. Coordina el equipo de trabajo y es responsable de la planificación y seguimiento del proyecto. Es el interlocutor del equipo de trabajo con el profesor supervisor del caso de estudio y presenta los resultados del proyecto.
- **Responsable de Gestión de Configuración**. Propone y pone en práctica procedimientos y herramientas para la evolución controlada del sistema en desarrollo. Es responsable de la creación y mantenimiento de la *baseline* de proyecto e integrador del sistema final y las versiones intermedias.
- **Responsable de Análisis**. Interpreta y completa los requisitos del sistema y los formaliza en un conjunto de modelos de análisis. Como resultado final obtendrá la especificación de requisitos del sistema.
- **Responsable de Diseño**. Parte de la especificación de requisitos del sistema para derivar un diseño software del sistema a desarrollar. Este diseño formaliza la estructura del sistema con suficiente detalle como para permitir su realización física. Con frecuencia el responsable del diseño es también el responsable de la implementación del sistema.

Una vez que cada equipo de trabajo tiene asignado proyecto, comienza su desarrollo propiamente dicho. Las prácticas consisten en la planificación, gestión, seguimiento y realización del proyecto de desarrollo asignado mediante el ciclo de vida y el conjunto de técnicas, notaciones, herramientas y estándares propuestos para el caso.

Respecto del ciclo de vida, se han propuesto diferentes alternativas a lo largo de los últimos años: ciclo en cascada, modelo de proceso unificado, procesos ligeros, etc. Sin embargo, en los últimos cursos se ha optado por una versión ligera de proceso iterativo inspirada en el Modelo de Proceso Unificado.

Los alumnos presentan el trabajo realizado mediante un conjunto de entregas y presentaciones a lo largo del semestre. Estas entregas consisten en documentos técnicos y de gestión, modelos de análisis y diseño, código fuente, configuraciones software y, en general, cualquier producto intermedio o final relevante del proceso de desarrollo de la aplicación software. El plan de entregas depende de ciclo de vida asignado al equipo de trabajo y del plan de proyecto elaborado por el propio equipo al comienzo del semestre.

En cuanto a las presentaciones, se realizan una o dos presentaciones intermedias ante profesores y resto de alumnos, mediante las que los equipos de trabajo realizan un informe de progreso del proyecto, justifican decisiones de diseño o selección de herramientas y tecnologías y tratan de convencer de las ventajas de su sistema y de la adecuación de su enfoque de realización. Hay también una presentación final de la asignatura que incluye además una demostración de la aplicación o servicio desarrollado. Se trata en general de un prototipo operativo en base a los casos de uso principales del caso propuesto.

La evaluación del *Laboratorio de Ingeniería del Software* es continua y se hace en base a las prácticas realizadas y las presentaciones que los equipos de prácticas hacen a alumnos y profesores a lo largo del semestre.

En la presentación final, los alumnos actúan de evaluadores de las presentaciones de todos los grupos de su aula a excepción del grupo propio. Para ello completan una hoja de evaluación que entregan al final de cada sesión. La plantilla de evaluación disponible orienta a los alumnos sobre los diferentes aspectos a valorar: objetivos, estructura y contenidos, expresión, uso de recursos didácticos, ritmo y duración, interacción con la audiencia, demostración, etc.

4. EL ENTORNO DE HERRAMIENTAS DE SOPORTE AL APRENDIZAJE

El Departamento de Ingeniería de Sistemas Telemáticos de la UPM dispone de dos laboratorio docentes con una capacidad conjunta de unos 160 puestos de prácticas. Cada puesto cuenta con un PC que puede arrancar un sistema operativo LINUX o Windows (también hay máquinas con Mac OS X). La imagen del sistema operativo puede recargarse desde un servidor de binarios disponible en la red del laboratorio siempre que el usuario lo necesite. Esta imagen incluye, además del propio sistema operativo, un conjunto de herramientas comunes a las asignaturas del departamento: entorno ofimático, IDEs, entorno de ejecución Java, etc.

La red del laboratorio cuenta con varios servidores de propósito general para su gestión y operación: Servidor de Administración, Servidor Web, Servidor Moodle, Servidor de control de acceso remoto, Servidores de binarios, Router (Proxy/firewall), etc.

El servidor de acceso remoto permite el acceso autenticado a la red y servicios del laboratorio desde el exterior (vía Web y VNC). De hecho, este tipo de acceso es el más frecuente desde hace varios años, ya que la gran mayoría de los alumnos dispone de ordenadores personales y acceso a Internet de banda ancha, pudiendo trabajar en las prácticas desde cualquier lugar e incluso instalando localmente todo el entorno necesario y prescindiendo por tanto de los recursos del laboratorio docente en la UPM.

Al margen de estos servicios comunes, cada asignatura del departamento que lo precise puede contar con un servidor virtualizado para gestionar cuentas de grupo o despliegue de herramientas y entornos de ejecución específicos. Este es el caso del *Laboratorio de Ingeniería del Software*, que cuenta con un servidor (Odín) donde se instala el entorno específico de desarrollo y despliegue necesario para los proyectos propuestos. Estos componentes varían cada curso en función de la evolución tecnológica y del tipo de casos de uso propuestos. En los últimos cursos los componentes desplegados en Odín son los siguientes:

- *Entorno de desarrollo y ejecución*: entorno Java, IDE Eclipse (con *plug-ins* específicos), Subversion, Apache, Tomcat, Axis, MySQL, PHP5, etc.
- *Componentes y emuladores*: SMS-C, MMS-C, Emulador de servicios de posicionamiento (Ericsson MPS), eyeOS (*desktop web* para servicios *cloud computing*), etc.
- *Emuladores de sistema operativo para terminales celulares*: java ME toolkit, Android, etc.

También se cuenta con licencias de educación de algunas herramientas comerciales de soporte a la ingeniería del software como IBM RSA (UML) e IBM Doors (Ingeniería de requisitos).

En resumen, el laboratorio cuenta con los siguientes recursos específico: un sitio Web con información pública y restringida, una comunidad en la plataforma Moodle del Departamento y un servidor con un entorno específico para el desarrollo y despliegue de servicios telemáticos.

Este entorno se completa con una herramienta de desarrollo propio que soporta el trabajo colaborativo en torno a un proceso de desarrollo de software inspirado en el Modelo de Proceso Unificado. Este herramienta se describe en la siguiente sección.

5. APIS, UNA HERRAMIENTA PARA EL APRENDIZAJE COLABORATIVO DE LA INGENIERÍA DEL SOFTWARE

El Modelo de Proceso Unificado [6] es una propuesta de proceso de desarrollo vinculada al Lenguaje Unificado de Modelado (UML) y elaborada inicialmente por la misma empresa (Rational, hoy día adquirida por IBM) y equipo de trabajo que propuso este lenguaje de modelado. Se trata de un proceso iterativo y dirigido por riesgos que se basa en una sucesión de mini-proyectos (iteraciones) que refinan y completan incrementalmente el sistema en desarrollo hasta su versión final de producción. Este conjunto de mini-proyectos puede verse como sucesivos recorridos por las secuencia de actividades de un ciclo de vida en cascada

Desde la revisión docente de la asignatura *Laboratorio de Ingeniería del Software* en el curso 2005-06 en el contexto del proyecto de asignatura piloto para convergencia en el EEES, se optó por una versión ligera del Modelo de Proceso Unificado como proceso básico a utilizar por los grupos de prácticas del laboratorio. Por ligero, entendemos una adaptación del modelo de proceso general que selecciona sólo los artefactos más relevantes para documentar el sistema en desarrollo. Además, los alumnos sólo desarrollan las iteraciones correspondientes a las fases de *Inicio* y *Elaboración* con los siguientes objetivos:

- *Fase de Inicio*: obtener una visión global de los objetivos y requisitos del sistema y determinar la viabilidad técnica y económica del proyecto.
- *Fase de Elaboración*: Captura iterativa de la mayor parte de los requisitos del sistema, determinación de la arquitectura del sistema y desarrollo incremental de ésta en base a los casos de uso principales del sistema.

Además de estos objetivos principales, los alumnos realizan tareas vinculadas a las disciplinas de *Gestión de Proyecto*, *Gestión de Configuración* y *Entorno*.

El hito que en el caso de un proyecto de desarrollo real se correspondería con el fin de la fase de *Elaboración* se hace coincidir con el fin del trabajo académico para los alumnos, momento en que los alumnos realizan una demostración de un prototipo operativo que implementa los casos de uso principales del sistema. En general, la mayoría de los grupos realiza el proyecto en base a una o dos iteraciones en la fase de *Inicio* y dos o tres iteraciones en la fase de *Elaboración*.

El *Laboratorio de Ingeniería del Software* cuenta con una aplicación Web de soporte a la versión adaptada del Modelo de

Proceso Unificado. Esta aplicación (APIS) permite definir proyectos, participantes, roles, iteraciones, artefactos y entregas y es utilizada por los miembros del proyecto para adaptar el modelo de proceso a su caso de estudio, registrar los documentos elaborados y realizar entregas: conjunto de artefactos que constituyen un compromiso de entrega en un hito de proyecto.

La aplicación APIS permite el trabajo colaborativo remoto de los miembros del proyecto y el seguimiento por parte del profesor supervisor. En su implementación inicial, APIS era un *front-end* o aplicación cliente de BSCW, herramienta de trabajo colaborativo que se utilizaba para soportar el trabajo en grupo y la gestión documental de las entregas realizadas.

Esta herramienta se ha utilizado con éxito a lo largo de varios cursos en la impartición del laboratorio y en el curso 2009-10 ha sido rediseñada completamente con objeto de mejorar su funcionalidad y prestaciones y hacerla autocontenida, de forma que APIS pueda funcionar de forma autónoma o con otras plataformas de trabajo colaborativo o de gestión de aprendizaje, en particular Moodle.

En las siguientes subsecciones se ofrece una descripción más técnica de la funcionalidad, arquitectura y tecnologías de implementación de la aplicación rediseñada.

5.1 Requisitos funcionales

Como ya se ha descrito, el objetivo principal de APIS es proporcionar a los usuarios la capacidad de realizar todas las tareas y gestiones involucradas en un proyecto de desarrollo software, guiado por la metodología propuesta en la asignatura *Laboratorio de Ingeniería Software*.

Las tareas y gestiones concretas que debe proporcionar el sistema dependen del tipo de usuario que haga uso de él, así como del perfil que utilice dentro del sistema.

Actualmente existen tres tipos de usuarios:

- **Alumno**: En este conjunto se incluyen todos los alumnos matriculados en el laboratorio. Podrán acceder al sistema con dos perfiles distintos: *grupo de trabajo* o *usuario* individual. Estos usuarios utilizarán la aplicación para completar el proyecto asignado a su equipo de trabajo. Los roles que puede asumir cada miembro del equipo han sido detallados en la sección 3.
- **Profesor**: En este conjunto se incluyen todos los profesores y personal docente que impartan el laboratorio. Podrán acceder al sistema con dos perfiles distintos: *supervisor* o *administrador*. Estos usuarios accederán a la aplicación para realizar un seguimiento tanto del progreso de los alumnos asignados a sus proyectos como del funcionamiento general de la aplicación.
- **Personal autorizado**: En este grupo se incluyen aquellas personas que han recibido autorización por parte del personal docente del laboratorio para acceder al sistema con el perfil *administrador*.

La forma más intuitiva de reflejar los requisitos funcionales gráficamente es a través de un diagrama de casos de uso.

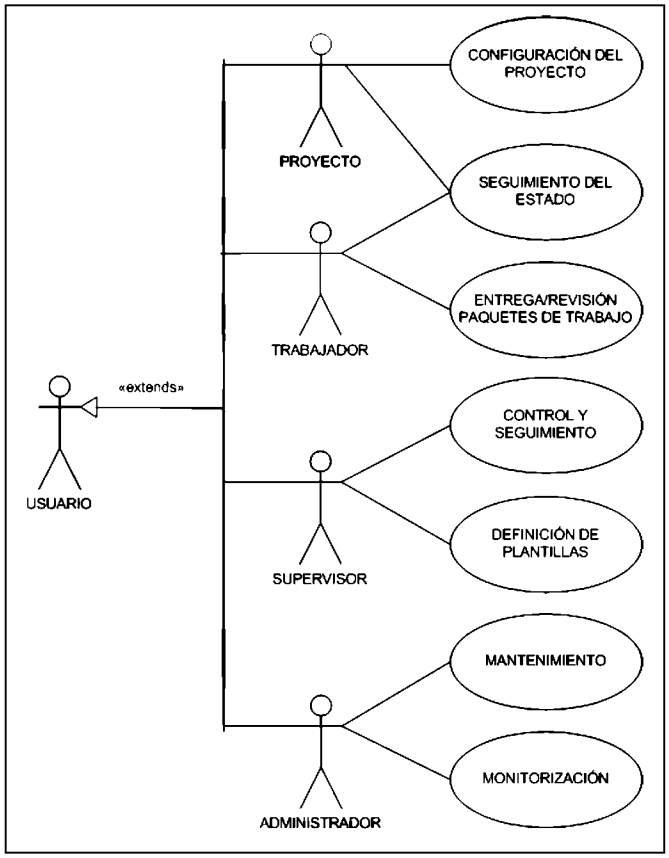

Figura 2. Diagrama de casos de uso

En el diagrama pueden apreciarse cuatro actores distintos. Estos actores se han generado en base a los perfiles de uso identificados más arriba para cada uno de los tipos de usuario de la aplicación.

Las características de cada uno de estos actores pueden resumirse a continuación:

- **Proyecto:** Este actor representa al grupo de trabajo asignado a un proyecto de desarrollo software concreto. A través de este actor pueden acceder a la definición inicial y al estado actual del proyecto todos los usuarios pertenecientes a un grupo de trabajo.
- **Trabajador:** Este actor aglutina todos los roles propios de un equipo de desarrollo software: Jefe de proyecto, responsable de integración, Analista programador y diseñador. A través de él, es posible que cada integrante de un equipo de desarrollo pueda realizar aquellas tareas que le han sido asignadas.
- **Supervisor**: Este actor representa a los profesores de la asignatura, que utilizan la aplicación para proponer casos de estudio y supervisar su desarrollo.
- **Administrador**: Este actor representa a aquellos usuarios encargados del mantenimiento y supervisión de la aplicación. Sus funciones son independientes del desarrollo de los proyectos establecidos.

La siguiente figura muestra, a modo de ejemplo, la interfaz de la aplicación para el perfil supervisor.

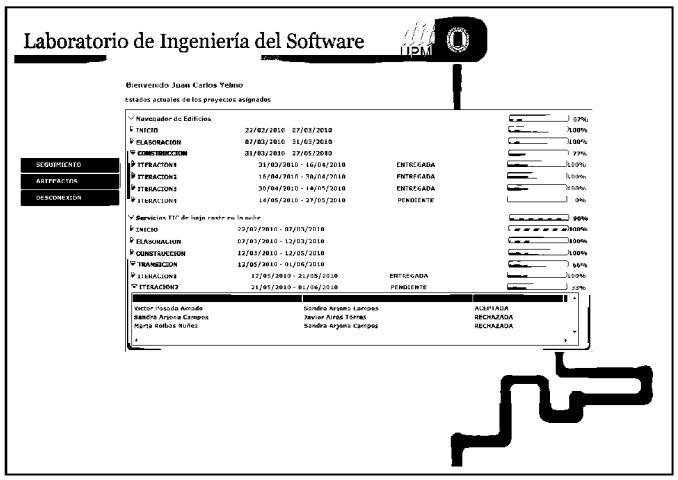

Figura 3. Pantalla de control de proyectos de un Supervisor

5.2 Requisitos no funcionales

A continuación se describen algunas conclusiones del análisis de requisitos respecto de los requisitos no funcionales más relevantes.

Seguridad

Los requisitos prioritarios en materia de seguridad tienen que ver fundamentalmente con los mecanismos de control de acceso (autenticación y autorización) y gestión de sesiones. En este sentido, se ha propuesta un esquema de control de acceso basado en roles (RBAC, *Role-Based Access Control*). Este esquema está basado en la creación de un conjunto de roles dentro de la estructura organizativa de los usuarios que accederán al sistema. Cada uno de estos roles representa un conjunto de funciones que pueden llevar a cabo los individuos pertenecientes a este rol. Para cada una de las operaciones que hay en el sistema se define el conjunto de roles que tienen permiso para realizarla.

Como gestor de autenticación se hace uso del servidor LDAP de la UPM, el cual tendrá definidas sus políticas de reintentos y bloqueo de usuarios. De esta manera la aplicación no almacena ninguna de las contraseñas que son usadas para acceder a la aplicación, por lo que cualquier actualización o cambio de las mismas es transparente para el sistema.

Otro aspecto fundamental de seguridad consiste en la gestión que realiza tanto la aplicación como el contenedor de *servlets* de las sesiones que actualmente están abiertas. Para cada petición recibida se realizan dos validaciones, validez de la sesión y posible robo de sesión. Estos robos pueden realizarse si un usuario malintencionado consigue hacerse con el identificador de sesión que ha establecido el servidor para ese usuario.

Usabilidad

Este requisito afecta particularmente al diseño de la interacción de los usuarios con la herramienta de gestión del laboratorio de tal manera que la captura de datos, la navegación, la administración, el mantenimiento y la inserción de documentos sean naturales y ajustadas a las necesidades de los usuarios finales de la aplicación. En este sentido se ha cuidado especialmente el diseño de páginas, la disponibilidad de ayuda on-line, tanto global como contextual, y la internacionalización completa de la aplicación.

Robustez

Los factores fundamentales a tener en cuenta en este sentido son la disponibilidad y el funcionamiento correcto de la aplicación ante picos de uso intensivo en cuanto a número de usuarios y transferencia de grandes volúmenes de información. En el caso de la disponibilidad, se cuenta básicamente con los mecanismos de mantenimiento generales del Centro de Cálculo del departamento: monitorización, reinicios programados, política de back-ups, etc. Para validar el comportamiento de la aplicación ante su uso intensivo se han realizado pruebas de stress (ver más adelante)

Mantenibilidad

La mantenibilidad es un requisito importante en cualquier proyecto y más aun cuando la aplicación no se desarrolla por parte de una empresa con personal, procedimientos, recursos y herramientas dedicadas a esta labor. En nuestro caso se trata de un desarrollo académico, que se lleva a cabo en gran medida por alumnos becados que dejan la Universidad al acabar sus estudios, por lo que la evolución de la aplicación se lleva a cabo por nuevos alumnos que no han participado en su implementación inicial. En este sentido, se ha cuidado especialmente el diseño de arquitectura, la documentación del código y la existencia de manuales y documentación técnica, de forma que sea viable la evolución del sistema con desarrolladores en permanente rotación.

5.3 Arquitectura y selección de tecnologías

En un primer nivel, APIS está diseñada siguiendo el patrón estructural *Modelo-Vista-Controlador* (MVC), impuesto por el *framework* base utilizado para el sistema.

Las capas en las que está estructurado el sistema son las siguientes:

- **Presentación**: Los componentes de esta capa, también conocida como *Vista*, tienen por función la realización del modelo de interacción con el usuario y la presentación de información. Como se puede comprobar en el esquema, existen dos circuitos distintos de peticiones: Síncronas y asíncronas. Las primeras son consecuencia de una acción consciente del usuario, que solicita al sistema una determinada acción. Las segundas se generan automáticamente, vía AJAX, sin que el usuario sea consciente de ello, y no modifican la vista entera, sino solo un segmento de la misma.
- **Aplicación**: Esta capa, también conocida como *Controlador*, aglutina todos los componentes encargados de controlar el flujo de datos y de atender las peticiones del usuario y ejecutar la lógica interna del sistema.
- **Datos**: También denominada *Modelo*. En esta capa residen todos los componentes encargados de interactuar con el mecanismo de persistencia de la aplicación, formada en este caso por una base de datos relacional.

Esta separación en capas hace de la aplicación un sistema modular, flexible y extensible, permitiendo la modificación de aspectos importantes del sistema sin tener que llevar a cabo una refactorización completa.

La arquitectura puede resumirse en el siguiente esquema visual:

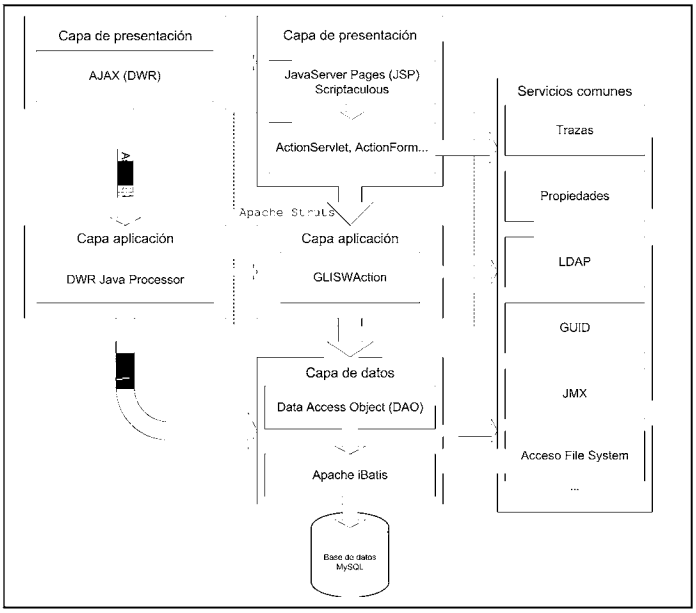

Figura 4. Esquema general de la arquitectura

Las tecnologías utilizadas en la implementación de la aplicación son las siguientes:

- Java (J2EE). Entorno de programación y ejecución de aplicaciones.
- Apache Struts. Framework de creación de aplicaciones Web basadas en Java.
- Apache iBatis. Framework de capa de persistencia para correspondencia objeto-relacional.
- Javascript y AJAX. Lenguajes de scripting para creación de aplicaciones Web interactivas.
- HTML y CSS. Formato de datos y hojas de estilo de presentación.
- Apache Tomcat. Contenedor J2EE para Servlets y JSPs.
- MySQL. Gestor de bases de datos relacionales.

5.4 Validación

Para asegurar en la medida de lo posible el correcto funcionamiento del sistema se han llevado a cabo una serie de pruebas sobre los componentes críticos del código, así como sobre el sistema global para verificar que cumple los requisitos.

Pruebas unitarias

Estas pruebas cubren todo el comportamiento de los componente de la capa de datos del sistema. Se encargan de comprobar que cada componentes de la capa de datos funciona correctamente por separado.

Estos componentes se encargan de las operaciones básicas que el sistema realiza sobre la información que gestiona. Para llevar a cabo estas pruebas se han codificado pruebas de todos los componentes de la capa de datos utilizando el *framework* JUnit.

Pruebas de stress

Estas pruebas pretenden obtener el comportamiento del sistema frente a situaciones en las que varios usuarios estén realizando acciones sobre él simultáneamente.

Se puede simplificar suponiendo dos comportamientos distintos por parte de los usuarios: Los que realizan una navegación normal sobre la aplicación, consultando y alterando datos del sistema, y los que manipulan ficheros del sistema, tanto subiendo nuevos ficheros como descargando los que ya contiene el sistema.

Para llevar a cabo estas pruebas se ha utilizado un software comercial llamado *Web Application Testing* (WAPT), configurándolo para que simule peticiones provenientes de veinte usuarios distintos sobre el sistema.

Las siguientes figuras muestran datos de consumo de memoria y CPU y tiempo de respuesta durante las pruebas de stress.

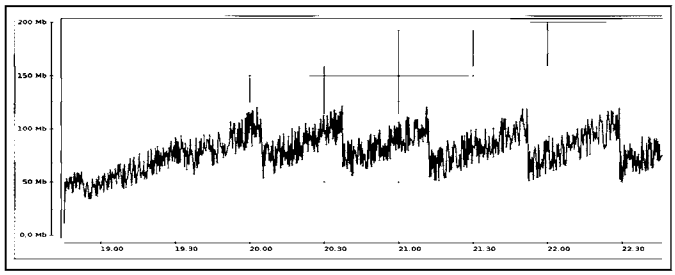

Figura 5. Consumo de memoria durante las pruebas de stress

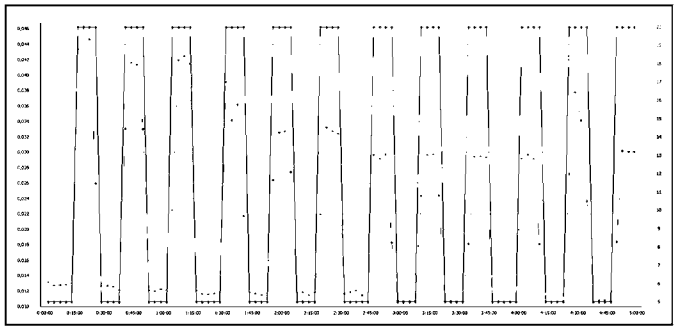

Figura 6. Tiempos de respuesta durante las pruebas de stress

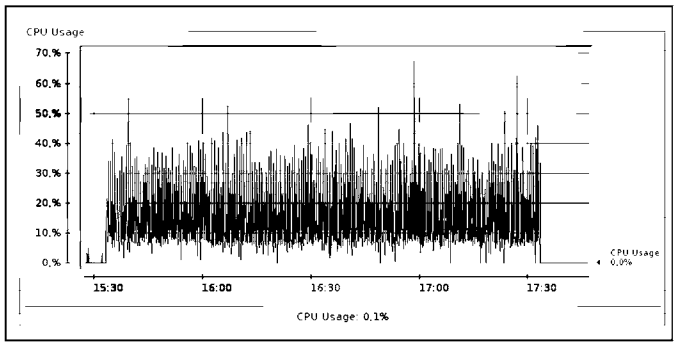

Figura 7. Consumo de CPU durante las pruebas de stress

En ambos casos (navegación normal y transferencia de ficheros) se ha comprobado que el sistema es capaz de asumir una carga muy alta de peticiones sin degradar los tiempos de respuesta del sistema ni los recursos consumidos por éste.

En cuanto a los tiempos de respuesta, el sistema ha demostrado tener un comportamiento más que suficiente, presentando unos tiempos de respuesta adaptados al número de peticiones concurrentes, pero sin degradar el rendimiento del sistema a niveles inaceptables.

6. CONCLUSIONES

En este artículo se ha descrito la experiencia de innovación educativa en torno al aprendizaje de la materia Ingeniería del Software en la especialidad de Telemática de la titulación de Ingeniero de Telecomunicación. En particular, se ha descrito el proyecto de adaptación de la asignatura *Laboratorio de Ingeniería del Software* a las nuevas directrices docentes emanadas de la declaración de Bolonia. En este contexto, se ha descrito en detalle la organización y metodología docente de la asignatura, el propósito y resultados de este proyecto de innovación educativa, el entorno de herramientas de soporte docente del Laboratorio y la herramientas APIS, como ejemplo de herramienta docente de soporte al aprendizaje colaborativo.

La experiencia nos ha permitido revisar y mejorar notablemente la eficacia docente de la asignatura en términos de organización, metodología y entorno de herramientas de soporte al aprendizaje, así como la medida de carga de trabajo real para alumnos y profesores para proponer una medida realista de esfuerzo en ECTS. En este sentido, las encuestas realizadas reflejan una media de dedicación declarada por los alumnos de unas 110 horas para superar la asignatura. Estas mismas encuestas reflejan una buena valoración sobre el grado de aprendizaje de conocimientos adquiridos en la asignatura y los métodos docentes empleados.

Con los parámetros habituales de cálculo, la carga de trabajo para el alumno se situaría por tanto en torno a 3,7-4,4 ECTS (en función de la habitual orquilla de valoración en horas de los ECTS). A partir de estas cifras, la carga docente real para el profesorado se sitúa en torno a las 50-55 horas.

Quedan muchos retos por delante y probablemente el más importante sea adaptar la formación de la materia de Ingeniería del Software a la nueva arquitectura de la titulación en base a los dos niveles académicos de grado y máster.

7. AGRADECIMIENTOS

Los autores quieren agradecer la contribución y dedicación a este proyecto de Mercedes Garijo y Miguel Ángel de Miguel, profesores del Laboratorio de Ingeniería del Software, de Pedro Clemente y Luis Mateos, primeros desarrolladores de APIS y del personal del Centro de Cálculo del DIT-UPM, siempre dispuestos a apoyar la evolución tecnológica de los laboratorios docentes. Finalmente, dar las gracias al Vicerrectorado de Ordenación Académica y Planificación Estratégica de la UPM por su apoyo y financiación de este proyecto de innovación educativa.

8. REFERENCIAS

[1] [1] C. Larman, Applying UML and Patterns. An introduction to Object-Oriented Analysis and Design and Iterative Development, 3ª edición, Upper Saddle River, NJ, USA: Prentice Hall PTR, 2004.

[2] [2] H. Van Vliet. Software Engineering. Principles and Practice. 3ª edición. Wiley, 2008.

[3] [3] ISO/IEC 12207:2008 "Systems and software engineering – Software life cycle processes," ISO JTC 1/SC 7, Mar. 2008.

[4] [4] A. Abran, J. W. Moore. Eds "Guide to the Software Engineering Body of Knowledge (SWEBOK)" IEEE computer Society, Mar. 2004.

[5] [5] J.T. Marchewka. Information Technology Project Management. Providing Measureable Organizational Value. 3ª Edición. John Wiley & Sons, 2009.

[6] [6] I. Jacobson, G. Booch, J. Rumbaugh. The Unified Software Development Proces. Addison-Wesle, 1999.