El Control de Versiones en el aprendizaje de la Ingeniería Informática: Un enfoque práctico

Fran J. Ruiz-Bertol, Francisco Javier Zarazaga-Soria
Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza
c/ María de Luna 1, 50018 Zaragoza
franjr@unizar.es, javy@unizar.es

Resumen

En la formación necesaria para el desempeño de las tareas de un Ingeniero en Informática es necesario conocer, aprender y usar herramientas para la gestión de proyectos. Esto debe llevarse a cabo haciendo uso de los recursos disponibles en el marco de la enseñanza universitaria, y sin olvidar el trasfondo de aprendizaje que hay más allá de la herramienta específica que se utilice. Una aproximación muy válida es la basada en el uso de recursos *open source* de éxito por dos razones: el alumno va a poder contar con ellos en su vida profesional (cosa que no ocurre con los productos comerciales); y suelen ser el resultado de la amalgama de buenas prácticas y experiencias de grupos muy heterogéneos. Uno de los aspectos fundamentales del proceso de aprendizaje en las habilidades de gestión de proyectos es tener un control sobre los cambios que se producen durante un proyecto, para facilitar la gestión, revisión y auditoria de tanto los productos o servicios software producidos como el proceso de cambios realizado. En este artículo se presenta el aprendizaje de un sistema de gestión de configuraciones desde un enfoque práctico, apoyado en una herramienta *open source* con dos objetivos primordiales: (i) conocer y aprender a usar una herramienta para la gestión de configuraciones y (ii) afianzar los conceptos de control de cambios, versiones y auditoria de una manera práctica.

1. Introducción

La sociedad demanda cada día más a profesionales universitarios que sean capaces de incorporarse al mercado laboral con plenas garantías de que su formación durante su carrera ha sido adecuada tanto a nivel teórico como práctico. Concretamente, para un Ingeniero en Informática se requiere una buena base teórica de aspectos algorítmicos, de diseño, de arquitecturas, capacidad de resolver problemas, capacidad de análisis, entre otros. Todo este conjunto de aspectos teóricos se cumplen de manera adecuada.

Desde un punto de vista práctico, un Ingeniero Informático debe ser capaz de desenvolverse y adaptarse a las situaciones o problemas que surgen durante su desarrollo profesional, así como aplicar técnicas, herramientas y habilidades aprendidas durante sus estudios. Esto implica que los futuros profesionales informáticos, no sólo deben tener los conocimientos teóricos de cómo resolver ciertas situaciones que pueden darse en el mundo profesional, sino que también deben estar dotados de un conocimiento práctico y experiencias que le permitan seleccionar en cada momento el conjunto de herramientas necesarias para la toma de decisiones.

En el libro blanco del título de grado en Ingeniería Informática [1], una de las competencias específicas fundamentales que es común para los tres perfiles sugeridos (Desarrollo Software, Sistemas, y Gestión y Explotación de las TI) es la Dirección, Planificación y Gestión de Proyectos. Para capacitar al alumno en esta competencia, es necesario desarrollar su aprendizaje desde un nivel eminentemente práctico durante el estudio de la carrera como competencia transversal, y formalizarla en los últimos cursos de la titulación para que ésta sea efectiva en su proceso de aprendizaje como Ingeniero. El actual plan de estudios de Ingeniería en Informática de la Universidad de Zaragoza (BOE de 1 de febrero de 1995) estructura la materia troncal de Ingeniería del Software en tres asignaturas: Ingeniería del Software I, Ingeniería del Software II y Proyectos. Para esta última asignatura se han reservado los contenidos directamente relacionados con la gestión de

proyectos. Una asignatura de estas características es parte del plan de estudios troncal común a todas las Universidades Españolas (BOE de 26 de octubre de 1990), y también corresponderá a una materia fundamental para el Espacio Europeo de Enseñanza Superior en el grado de Ingeniería Informática (BOE de 21 de enero de 2005). De manera genérica, la gestión de proyectos puede definirse como "*la aplicación de conocimientos, habilidades, herramientas y técnicas a las actividades del proyecto para cumplir los requisitos*" [2].

Como los resultados principales del desarrollo de un proyecto informático son fundamentalmente productos o servicios software (código fuente, aplicaciones, manuales, documentos de trabajo, etc.) es imprescindible mantener una coherencia e integridad de dichos entregables. Este fin se alcanzará utilizando, por una parte, los conocimientos y habilidades necesarios para avanzar en el cumplimiento de requisitos, y, por otra, utilizando las herramientas y técnicas necesarias para que dichos productos de trabajo guarden una coherencia e integridad adecuada. Por ello, es necesario que durante su periodo formativo, los alumnos sean capaces de adquirir los conocimientos y puedan desarrollar las habilidades para el desempeño de la profesión, pero que también que sepan utilizar las herramientas necesarias que faciliten el desempeño de su tarea. A la hora de plantear una práctica en este contexto, ha resultado complejo dar con otros ejemplos que nos guíen. Esto ha llevado a la necesidad de transmitir nuestra experiencia para el aprovechamiento de otros.

En este artículo, presentamos la implantación de un sistema de control de versiones *open source* denominado Subversion en la asignatura Proyectos de la Universidad de Zaragoza. La implantación de este software supone, no sólo instalar los programas y servicios necesarios para que el software funcione, sino que además se encuentra el objetivo de que los alumnos hagan uso de forma guiada de una herramienta que sirva para el control de configuraciones, mantenimiento de versiones y revisiones, aprendizaje de los problemas que pueden surgir en un entorno de acceso compartido, y la introducción de los procesos de registro y calidad en el desempeño de la profesión.

En la sección 2 se presenta una guía general del proceso de implantación de Subversion y su cliente gráfico Tortoise SVN, detallando el conjunto de funcionalidades que permite este software y su aplicación lógica en el entorno de un proyecto software. La sección 3 detalla la metodología de trabajo desde el enfoque de los objetivos de aprendizaje y verificación de los mismos que se espera de los alumnos durante el desarrollo de la práctica. Este trabajo termina con una sección de conclusiones.

2. Subversión y Tortoise SVN

La gestión de configuración del software (SCM) se define como "*el conjunto de actividades diseñadas para el control de cambios identificando los productos de trabajo susceptibles de cambiar, estableciendo las relaciones entre ellos, definiendo mecanismos para la gestión de diferentes versiones de esos productos de trabajo, y controlando, auditando y transmitiendo los cambios realizados*" [3]. Hasta hace pocos años, el software SCM por excelencia era *Concurrent Versioning System* (CVS), creado por Grune en los años 80 [4], y que implementaba bajo una arquitectura cliente/servidor el conjunto de funcionalidades necesarias para el control de configuraciones. CVS trabaja principalmente bajo entornos Unix, pudiendo conectarse a dicho servidor cualquier cliente que implementara el protocolo de comunicación establecido por CVS [5].

Sin embargo, en los últimos años, CVS ha quedado relegado a un segundo plano tras la aparición de Subversion [6], debido, principalmente, a que este último soluciona problemas que existían en CVS, como el mantenimiento de versiones tras renombrar o mover un archivo/directorio, una difícil integración de los conjuntos de caracteres de internacionalización, o una compleja estructura del árbol de directorios del sistema. En esta sección se proporciona una breve guía sobre Subversion, así como los pasos necesarios para la instalación, configuración e implantación de la arquitectura cliente/servidor a través del cliente Tortoise SVN [7].

2.1. Subversion

Subversion (SVN) es un sistema de control de configuraciones *open-source*. Esta herramienta de control de versiones también puede considerarse como un repositorio que guarda la información de archivos y directorios, incluyendo la información asociada a las modificaciones realizadas. Los clientes pueden acceder a SVN para solicitar archivos, actualizarlos, modificarlos e incluso crear nuevos ficheros y disponerlos para los demás usuarios en el repositorio. Pero la funcionalidad de SVN no se basa en ser un software que actúa de servidor de archivos, sino que incorpora las características necesarias para ser un SCM. Entre estas características se encuentra el mantenimiento de versiones y revisiones antiguas, la información asociada a cada una de las versiones o revisiones, la auditoría, la posibilidad de que varios usuarios estén trabajando sobre un mismo archivo y se contemplen sin bloqueos las modificaciones realizadas por cada uno de los usuarios, comparar las diferencias entre archivos de distintas versiones, y disponer de un almacén de datos que contenga la información de un proyecto dada actualizada. Este último hecho se convierte en fundamental para muchas empresas de desarrollo software, equipos virtuales y esfuerzos para el desarrollo de software libre, ya que en muchas ocasiones los implicados en el desarrollo deben disponer en un momento dado de archivos compartidos (y posiblemente sujetos a cambios de manera continua y concurrente) y poder realizar modificaciones que no afecten a los demás usuarios. Por ejemplo, supongamos un equipo virtual, distribuido en diferentes localizaciones mundiales y con distintos husos horarios. Si un usuario accede para modificar una función del código fuente de un proyecto software, y a su vez, otro usuario está consultando y refinando otro conjunto de funciones del mismo código fuente, a través de SVN ambos usuarios podrán confirmar correctamente sus modificaciones a través del comando *merge* sin afectar a las modificaciones realizadas por el otro usuario.

SVN está disponible para la mayoría de plataformas (Windows, Linux/Unix, Mac y Solaris) [10]. A efectos prácticos, en este artículo se desarrollarán las instalación, configuración y la utilización de las funciones más importantes en una plataforma Windows, ya que éste será el sistema operativo a utilizar en las prácticas asociadas a la asignatura de Proyectos.

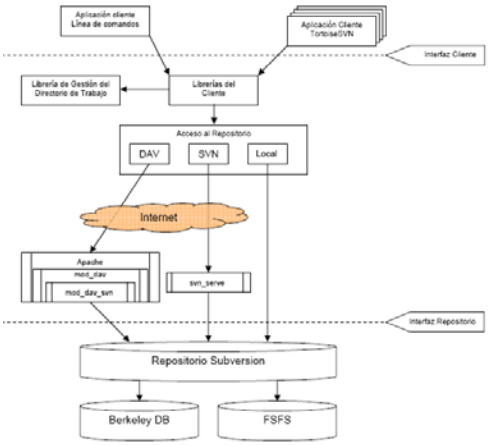

Figura 1. Arquitectura y Estructura de los módulos de Subversion obtenida de [8].

Primero es necesario obtener los archivos de instalación más adecuados para nuestros propósitos. SVN se ofrece tanto como código fuente (siempre bajo licencia *open-soruce*) como en binario o compilado para cada una de las plataformas. Una vez descargado, la instalación es completamente guiada, y dejará instalado SVN en el directorio seleccionado. Después, será necesario arrancar el servicio correspondiente para poder trabajar con repositorios. Sin embargo, esto implica que es necesario configurar adecuadamente el software, que dependerá de la arquitectura que deseemos para la utilización de SVN.

Desde el punto de vista de la conexión, SVN proporciona tres arquitecturas: local, usando el módulo `web_dav_svn` en Apache, y a través del servicio `svnserve` (ver Figura 1).

Para crear un repositorio SVN tanto en modo local como remoto, se utiliza el siguiente comando desde la consola: `svnadmin create "file:///Ruta/Repositorio"`, siendo este último parámetro el directorio donde vamos a crear el repositorio. Si dicho repositorio quiere exponerse para que actúe de manera remota, se instala como servicio dentro de Apache

incluyendo el módulo `web_dav_svn` en la configuración del servidor Apache, e indicando la ruta del repositorio recién creado [9]. Si disponemos de un servidor limitado, o no deseamos configurar el servidor Apache, SVN proporciona el servicio autónomo `svnserve`, que permite utilizar el repositorio de la misma manera que con el servidor.

Desde el punto de vista de almacenamiento de la información, los repositorios SVN pueden crearse sobre dos tipos de almacenes de datos: Berkeley DB [10] y FSFS [11]. Berkeley DB proporciona un almacén de datos escalable y un acceso rápido, soportando transacciones reales, copias de seguridad y coherencia en los datos. FSFS proporciona otro enfoque en el manejo de ficheros en el repositorio, aunque proporciona todas las características de Berkeley DB pero sin ser en el sentido estricto una base datos. FSFS, además, mejora las transacciones incluyendo en un único fichero de transacción todas las modificaciones de un usuario, reduce el tamaño del repositorio (en torno a un 20%), y es independiente de la plataforma [12].

Una vez creado el repositorio SVN, para utilizar dicho repositorio tanto de manera local como remota, es necesario una copia de trabajo usando el siguiente comando `svn checkout` repositorio. Este comando sirve tanto para crear una copia local de los archivos que hay en el repositorio, pero también para reemplazar y actualizar los ficheros existentes en la copia de trabajo.

Ya esa en modo local como remoto, es necesario establecer una autenticación de los usuarios que les permita acceder al repositorio SVN, y una autorización que permita a dichos usuarios acceder en modo lectura, lectura/escritura o permitir el acceso de dichos usuarios a ciertos recursos del repositorio SVN [8]. Esto se realiza a través de los ficheros de configuración que se encuentran en el directorio `/conf` del repositorio. Para establecer las políticas de autorización y autenticación, SVN dispone del fichero svnserve.conf, que determina los métodos montados sobre el repositorio para determinar los archivos de autenticación y autorización, proporcionando incluso métodos de configuración a través de SSH.

Entre las operaciones disponibles en SVN se destacan las siguientes:

- `create`. El comando `svn create directorio` crea un nuevo repositorio. Una vez este se crea, habrá que arrancar los servicios remotos adecuados (si es necesario) y establecer la configuración para la autenticación y autorización de los usuarios.
- `import`. El comando `svn import [origen] repositorio` permite realizar una importación inicial o volcado al repositorio de forma recursiva.
- `list`. El comando `svn list` permite listar el contenido del repositorio.
- `update`. Si no estamos seguros de si estamos trabajando con los archivos en su última versión en nuestra copia privada, es necesario que actualizar la copia de trabajo usando el comando `svn update repositorio`. Este comando comprueba si el contenido del repositorio y de la copia de trabajo son coherentes, y en caso contrario, transmite los ficheros contenidos en el repositorio a la copia de trabajo.
- `commit`. El comando `svn commit` confirma en el repositorio los cambios realizados en la copia de trabajo. Si existe conflicto de datos, nos avisa de ello.
- `add`. El comando `svn add` añade nuevos ficheros/directorios a la copia de trabajo. Sin embargo, estos cambios no son efectivos hasta que los ficheros o directorios son confirmados a través del comando `commit`.
- `delete`. El comando `svn delete` elimina de la copia de trabajo los archivos o directorios especificados como parámetro. El efecto del comando será efectivo cuando se realice la operación `commit`.
- `copy`. El comando `svn copy origen destino` copia el fichero o directorio origen al fichero o directorio destino a la copia de trabajo actual. En el repositorio se comprometerán estos cambios tras la operación `commit`.
- `move`. El comando `svn move origen destino` renombra el fichero o directorio origen al nombre especificado por destino. Los cambios serán efectivos en el repositorio tras la confirmación.
- `status`. El comando `svn status` determina el estado de la copia de trabajo en comparación con el contenido del repositorio.

El estado se determina a través de un carácter, que puede ser 'A' (*add*), 'D' (*delete*), 'M' (*modified*), R (replaced) ó '?' (*non versined*).

- `diff`. El comando `svn diff` compara y detalla por pantalla las diferencias entre las distintas versiones de los archivos de la copia de trabajo y del repositorio.
- `revert`. El comando `svn revert archivo` establece en la copia de trabajo la versión inmediatamente anterior del archivo especificado.

2.2. Tortoise SVN

Tortoise SVN [7] es una aplicación basada las bibliotecas SVN que actúa de cliente gráfico para el acceso al repositorio SVN, tanto si éste es remoto como si es local. Este cliente gráfico ha sido diseñado para su integración contextual en el Explorador de Windows proporcionando el acceso a la mayoría de funciones que proporciona SVN.

Desde Tortoise SVN se pueden realizar todas aquellas operaciones relacionadas con la administración del directorio de trabajo (añadir, eliminar, modificar, mover), comunicación con el repositorio SVN (checkout, commit, actualizar, revertir, bloquear), resolución de conflictos (ramas, fusionar, exportar, relocalizar) y la creación de repositorios locales SVN. Un ejemplo de este menú contextual puede observarse en la Figura 2.

Con Tortoise SVN se puede observar en cada momento de manera icono-gráfica el estado de los archivos (nuevo, actualizado, modificado, eliminado, movido, renombrado, fuera de versión), así como de la copia de trabajo. De esta manera, se permite trabajar de una manera transparente con los archivos contenidos en un directorio de trabajo, pero también ser consciente de que si ha habido modificaciones, nuevos datos, borrado o conflictos con los datos actuales, éstos se marcan para que el usuario sea consciente de las distintas operaciones que debe realizar para transmitir estos cambio al repositorio SVN mediante las operaciones correspondientes.

Este cliente para SVN también contiene las ventanas de diálogo asociadas a las operaciones para poder determinar las opciones asociadas a cada una de las operaciones, así como los avisos al usuario y notas de versión que proporciona Subversion.

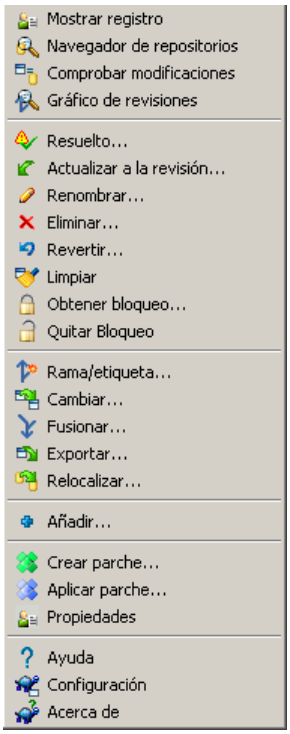

Figura 2. Menú contextual de Tortoise SVN integrado en el shell de Windows

3. Metodología de trabajo

El objetivo de conocer y utilizar este software viene dado a que la formación necesaria de los Ingenieros en Informática en las herramientas que facilitan la gestión de proyectos. De manera secundaria, a través de este software queremos que los alumnos aprendan el control de configuraciones, la gestión de calidad (a través de la anotación y seguimiento del proyecto durante su ciclo de vida) y conozcan el término auditoria tan importante para el desempeño profesional en entidades que poseen algún tipo de certificación de calidad (ISO 9001, Spice, CMMI, etc.).

El trabajo práctico se lleva a cabo en un entorno cliente-servidor sobre el sistema operativo Windows, donde primero se creará un repositorio en modo local, y luego se conectará a un servidor Solaris para probar las características de autorización y autenticación en modo remoto.

Como primer paso se procede a instalar por parte del profesorado Subversion en un servidor remoto y crear allí un repositorio SVN. En dicho repositorio se dejarán documentación y código fuente para que los alumnos puedan trabajar. Por otra parte, en cada uno de los equipos que vayan a utilizar los grupos de trabajo se instala el cliente Tortoise SVN, que les permita acceder a todas las funcionalidades para trabajar con un SCM.

Posteriormente, se definirán las reglas de autenticación y autorización de los grupos correspondientes con las cuentas de dominio asociadas, estableciendo permisos de lectura y escritura sobre los directorios que hemos dejado en el servidor, teniendo para cada grupo el acceso a un directorio propio, a un directorio compartido para todos, y varias combinaciones en modo lectura, y lectura/escritura para que puedan observar de una manera práctica las distintas operaciones que pueden realizarse con SVN. La documentación asociada a la realización de esta experiencia se encuentra publicada y disponible en [13]. Las operaciones que se desea que realicen los alumnos son las siguientes:

- Preparación del trabajo para un repositorio. Antes de crear los repositorios y las copias de trabajo, se debe realizar una preparación de la estructura del directorio de trabajo. Esta estructuración es principalmente crear una línea base de trabajo (`/trunk`), y los directorios para las ramas y etiquetas (`/branches` y `/tags`).
- Creación de un repositorio local. Se solicitará al alumno que cree un repositorio local usando el cliente Tortoise SVN, y se le proporcionará la documentación necesaria para que el alumno aprenda la arquitectura de un repositorio SVN, y cómo está estructurado éste para poder actuar de repositorio.
- Importación inicial. Una vez que están preparados el repositorio y el trabajo a importar, se realiza la importación a través del comando Importar de SVN.
- Exploración del repositorio. Una de las funciones más interesantes es comprobar el contenido del repositorio, ya que desde la ventana del navegador del repositorio nos proporciona un conjunto de operaciones de copiar, mover, eliminar y añadir nuevos ficheros, entre otras.
- Creación de una copia de trabajo. Se solicitará al alumno que cree un nuevo directorio de trabajo que se asocie al repositorio SVN recién creado a través del checkout. Esto dará lugar a la copia de trabajo local. Con esta tarea el alumno debe ser capaz de comprender la diferencia entre un repositorio, un directorio de trabajo y una copia de trabajo.
- Modificación de ficheros. Se ha proporcionado al alumno un sencillo código fuente en Java para la conversión de euros a libras. Para esta tarea será necesario que los alumnos trabajen sobre su copia de trabajo en la acción. Para comprobar las diferencias entre el archivo anterior y el que han tocado, podrán utilizar el comando diff.
- Comprobación de modificaciones. Es interesante que el alumno conozca qué modificaciones hay entre la copia de trabajo y el repositorio.
- Creación de nuevos ficheros. Se solicitará al alumno que cree en su copia privada nuevos archivos en la copia privada. Para ello, la manera más sencilla de crear nuevos archivos es solicitarles compilar (`javac`) y generar la documentación (`javadoc`) para el código fuente proporcionado.
- Actualización y comprometer los cambios. Estas dos operaciones son fundamentales para que se puedan confirmar los cambios realizados por los alumnos sobre el repositorio local, así como para que incluyan cada vez que comprometan estos cambios, las notas y mensajes asociados a la versión o revisión correspondiente.
- Resolución de conflictos. Se pondrá al alumno frente a un conflicto de datos, utilizando para ello un directorio compartido remoto donde se creó el repositorio. Su principal función será ser capaces de resolver este conflicto usando para ello las distintas copias existentes del archivo (revisión de donde se obtuvieron los datos, revisión actual del repositorio y copia local).
- Modificación concurrente de ficheros (opciones merge). Se solicitará a distintos grupos trabajen sobre el mismo fichero, modificando distintas partes del código fuente de un archivo. Posteriormente se solicitará que cada uno de ellos comprometa los datos, y

trabajen con la operación `merge` para hacer efectivos los cambios de ambos grupos.
- Auditar los cambios realizados sobre todo el repositorio. Los alumnos deben ser capaces de auditar los cambios realizados sobre el repositorio SVN, y observar el proceso que se ha seguido para obtener la configuración ó versión actual.
- Conexión remota y trabajo concurrente. Se solicitará al alumno que trabaje sobre un repositorio remoto donde tenemos arrancado un servidor SVN, y que realice operaciones de consulta y modificación.
- Bloqueo de archivos. Finalmente, se observará la efectividad de la operación de bloqueos, y cuándo debe utilizarse ésta.

De esta manera, cada uno de los alumnos conocerá las opciones básicas de la utilización de un SCM, y que son comunes a todos los sistemas de control de versiones. Su aprendizaje es fundamental no sólo como contenido exclusivo de la asignatura Proyectos, sino que, una vez adquirido el conocimiento de cómo utilizar esta herramienta, también han aprendido a utilizar una materia básica para poder gestionar los proyectos, adquiriendo diferentes roles: administrador, director, programador, gestor de proyectos, etc.

Esta experiencia docente tiene el objetivo de que los alumnos puedan aplicar lo aprendido durante el desarrollo de la práctica al desarrollo de su proyecto fin de carrera, y posteriormente a su carrera profesional, proporcionando así un valor añadido al finalizar su periodo de formación.

4. Experiencia práctica

A la hora de remitir la versión definitiva de este artículo, los alumnos han desarrollado esta experiencia docente de una manera eficaz, destacando en su gran mayoría la utilidad de aplicar esta experiencia de aquí en adelante, tanto en lo restante de sus estudios, como en su desarrollo profesional.

Los alumnos que han desarrollado esta práctica han sido conscientes de los problemas que surgen al trabajar con un sistema de gestión de configuraciones, pero cómo este hecho facilita la coordinación de los equipos de trabajo y la seguridad que proporciona disponer de un repositorio centralizado para los archivos de trabajo. De hecho, muchos alumnos han evidenciado que durante sus estudios han borrado alguna vez de manera accidental parte de sus prácticas, sin opción de poder recuperarlas.

Durante el desarrollo de la práctica, lo más complicado ha sido familiarizarse con las operaciones de Subversion y Tortoise SVN, ya que una gran minoría había trabajado anteriormente con sistemas de control de versiones. Tras comprender la documentación entregada y utilizarla de manera práctica, han adquirido dicho conocimiento.

5. Conclusiones

En este artículo se ha querido exponer una experiencia práctica de cómo aplicar la gestión de configuraciones durante la formación final de un Ingeniero en Informática. Esta experiencia ha surgido de la necesidad del compromiso de que la Universidad forme de manera adecuada a los alumnos, estando supeditado dicho aprendizaje a su asociación a cada materia docente impartida. En este caso concreto, la asignatura Proyectos debe formar en las capacidades de dirección, gestión, control y auditoria sobre los proyectos software.

Sin embargo, nos hemos visto en el problema de que en muchas Universidades Españolas no se encuentran experiencias docentes de este tipo o no han sido publicadas. Creemos fehacientemente que dentro de las competencias que deben adquirir los futuros Ingenieros en Informática es precisamente conocer y saber controlar las versiones de todos los productos/servicios que generen, independientemente de que a nivel corporativo se tenga previsto un proceso de copia de seguridad. Pensamos que esta experiencia resulta interesante de compartir con otros compañeros que se hayan fijado un objetivo docente similar al nuestro.

Así mismo, está a disposición de aquellas personas que lo soliciten el contenido de la práctica realizada [13]. Para ello, se recomienda que contacten con los autores de este artículo.

Referencias

[1] ANECA. *Libro Blanco: Título de Grado en Ingeniería Informática*. ANECA, 2005.

[2] Project Management Institute. *A guide to the Project Management Body of Knowledge: PMBOK guide*. 3rd. Ed. Project Management Institute, 2004.

[3] Presman, RS. *Software Engineering: A Practicioner's Approach*. 6th Ed. McGraw-Hill, 2005.

[4] Grune D. *Concurrent Versions System, a method for independent cooperation*. Internal Report 113, Vrije Universiteit, Amsterdam, pp. 9, 1986.

[5] Fogel, K. Bar, M. *Open Source Development with CVS*. 3rd Ed. Paraglyp, 2003.

[6] Tigris.org. *Subversion.* Electronic Source: http://subversion.tigris.org/ [Último acceso: 13/02/2007]

[7] Tigris.org. *Tortoise SVN*. Electronic Source: http://tortoisesvn.tigris.org/ [Último acceso: 13/02/2007]

[8] Collins-Sussman, B. Fitzpatrick, BW. Pilato, CM. *Version Control with Subversion*. O'Reilly, 2004.

[9] Clemm G. *et al*. *RFC 3253: Versioning Extensions to WebDAV (Web Distributed Authoring and Versioning)*. WebDAV Standards Track, Proposed Standard. 2002.

[10] Oracle Berkeley DB. *Berkeley DB*: Oracle Embeded Database. Electronic source: http://www.oracle.com/database/berkeley-db/db/index.html [Último acceso: 13/02/2007]

[11] FSFS. *The Fast Secure File System*. Electronic Source: http://fsfs.sourceforge.net/ [Último acceso: 13/02/2007]

[12] FSFS. *Subversion FSFS implementation: FSFS filesystem type repository*. Electronic Source: http://svn.collab.net/repos/svn/trunk/notes/fsfs [Último Acceso: 13/02/2007]

[13] Ruiz-Bertol, FJ; Zarazaga, FJ. *Control de Configuraciones. Control de Versiones con Subversion*. Educational Report ER-04-07. Dpto. Inf. e Ing. Sistemas. Univ. de Zaragoza, 2007.