

## **Aspectos docentes de la asignatura de Proyectos de Software del plan de estudios de Ingeniería Informática de la Universidad de Zaragoza**

Pedro R. Muro Medrano  
Javier Zarazaga Soria  
Juan Valiño García

Departamento de Informática e Ingeniería de Sistemas  
Centro Politécnico Superior  
Universidad de Zaragoza  
María de Luna 3  
50015 Zaragoza

### **Resumen**

*En este trabajo se presentan los aspectos docentes de la asignatura de Proyectos. La asignatura está planteada desde una perspectiva práctica en la que el trabajo del alumno se centra en el desarrollo de un proyecto completo de software. En el trabajo a realizar en el proyecto prevalecen aspectos de proceso, gestión y documentación y estándares sobre aspectos más técnicos del desarrollo como técnicas de análisis, diseño y aspectos de programación.*

### **1. Contexto de la asignatura**

El presente trabajo se centra en algunos aspectos docentes relacionados con la asignatura de Proyectos. Esta es una asignatura troncal del plan de estudios de Ingeniería Informática que se imparte en el Centro Politécnico Superior de la Universidad de Zaragoza.

La mencionada asignatura tiene una carga lectiva de 6 créditos totales de los cuales 3 son teóricos y otros 3 prácticos. Se encuentra situada en el octavo cuatrimestre de los estudios de Ingeniería Informática, constituyendo la última asignatura de las tres en que está dividida la materia troncal de Ingeniería del Software. Las otras dos asignaturas, con un total de 12 créditos (7,5 teóricos y 4,5 prácticos), son impartidas en el sexto (Ingeniería del software 1 con 3 + 1,5 créditos) y séptimo cuatrimestres (Ingeniería del software 2 con 4,5 + 3 créditos). Por otra parte, habría que resaltar también que el equipo de profesores que suscribe este trabajo es el primer año que imparte esta asignatura.

### **2. Objetivos del curso**

En las anteriores asignaturas del bloque temático de Ingeniería del Software el alumno ha recibido formación sobre una panorámica de las distintas fases del ciclo de vida del software y acerca de los

problemas, métodos y herramientas que pueden ser utilizadas en cada fase (como análisis de requisitos, diseño, producción, tests y mantenimiento). El planteamiento de esta asignatura es completar la formación del alumno en los aspectos técnicos y profesionales de la ingeniería del software y adquirir experiencia práctica en el desarrollo en equipo de proyectos de software. Dentro de este contexto general nos planteamos de forma más particular el trabajo en los siguientes aspectos:

- Estudio de problemas, métodos y herramientas relacionado con las actividades de gestión que se prolongan a lo largo de todo el ciclo de vida del software: gestión del proyecto, gestión de configuraciones y aseguramiento de calidad.
- Conocer el papel y uso de los estándares de software.
- Adquirir experiencia práctica en el desarrollo completo de proyectos de software donde se cubran todas las actividades de desarrollo y gestión y se contemple un producto de software, no sólo en su componente de código, sino conjuntamente con toda la documentación de producto asociada.
- Sensibilizar al alumno en la necesidad (que surge en muchas ocasiones) del trabajo en equipo y toda la problemática que esto lleva consigo. Para ello el alumno deberá enfrentarse a los problemas de organización y comunicación de información técnica y organizativa entre los miembros del equipo. Aspectos como definición de papeles de trabajo, responsabilidad en la realización de sus tareas y comprender el problema de dirección de un equipo en toda su extensión, son también elementos importantes que tratarán de cubrirse en esta asignatura desde una perspectiva fundamentalmente práctica.
- Hacer que el alumno tenga que elegir y poner en práctica el uso de técnicas aprendidas en asignaturas anteriores (y no sólo de ingeniería del software) para resolver o documentar distintos aspectos de un proyecto de software.

### 3. Contenidos

A continuación exponemos un extracto de los contenidos teóricos de la asignatura:

Contexto de la Ingeniería del Software  
Actividades y ciclo de vida del software  
    Fases, tareas y actividades  
    Ciclos de vida  
    Estándares  
Gestión del proyecto de software  
    Acuerdo del proyecto  
    Plan de Gestión del Proyecto de Software  
    Organización del proyecto  
    Proceso de gestión  
    Proceso técnico  
    Paquetes de trabajo, calendario y presupuesto  
Gestión de configuraciones de software  
    Plan de Gestión de Configuraciones de Software  
    Identificación de configuraciones  
    Control de código y documentación  
    Control de medios  
    Control de cambios  
    Contabilidad de estado de las configuraciones  
    Herramientas, técnicas y métodos  
Aseguramiento de la calidad del software

- Plan de Aseguramiento de la Calidad del Software
- Estándares, prácticas, convenciones y métricas
- Revisiones y auditorías
- Normas de calidad ISO 9000
- Modelos de madurez del software
- Medidas del software
  - Medidas de tamaño
  - Estimación del tamaño
  - Estimación de esfuerzos/costes
  - Medidas de productividad
  - Medidas de calidad

#### **4. Métodos de enseñanza utilizados**

Hay algo de lo que estamos absolutamente convencidos y es que se aprende con la experiencia. Esto es especialmente importante en una asignatura en la que los aspectos más teóricos como métodos, herramientas, formas de documentos, estándares, etc. tienen un carácter muy empírico y su explicación o análisis resulta frecuentemente aburrido. En este sentido creemos que no resulta rentable pedagógicamente que el alumno se aprenda de memoria gran cantidad de información, que por otra parte suele cambiar periódicamente (normalmente los estándares tienen una vigencia de 5 años), si no que sepan hacer uso práctico de estas informaciones apoyándose en la documentación y bibliografía necesarias.

Para poder llevar a cabo los objetivos propuestos (desglosados en la sección 2) hemos utilizado una estrategia docente que se puede desglosar en las siguientes partes:

- Una serie de clases teóricas, soportadas por transparencias, en las que se estudian los problemas, métodos y herramientas. En estas clases se usan los estándares (básicamente de la ESA y el IEEE) como guía para el desarrollo de los temas. En estas clases el alumno tiene un papel más pasivo, que sin embargo, se ve activado circunstancialmente al analizar en clase las propuestas de documentos de su proyecto.
- Unas prácticas en las que se usan herramientas o calculan medidas del sistema. Este aspecto práctico tiene incuestionable utilidad docente. Sin embargo debido a problemas de tiempo, de disponibilidad de herramientas y de no disponer de material preparado, estas prácticas no cubren todo el espectro posible del curso. (Esta es una de las partes que nos interesaría ampliar).
- Un proyecto de asignatura en la que se desarrolla completamente un sistema de software en equipos de trabajo de 4 a 6 personas en las que uno de los alumnos tiene el papel de director del proyecto. Este proyecto representa la carga más importante de trabajo de la asignatura para el alumno y constituye la base para su evaluación. El proyecto es de reducido tamaño y no tiene problemas técnicos, pero está obligado a seguir un proceso de desarrollo de razonable calidad. En este proyecto es donde adquiere experiencia sobre responsabilidades, colaboración, elaboración de documentos, conocimiento del proceso y gestión. Para ayudar a los equipos de desarrollo en esta tarea y asegurarnos de que van realizando los trabajos de forma aceptable sin tener que esperar al final del proyecto hemos introducido dos mecanismos:
  - Asesoría del proceso de desarrollo. Los equipos 1) reciben guías y plantillas para realizar sus propios documentos y planes de gestión, 2) disponen de un sistema de comunicación para preguntar cuestiones relacionadas con aspectos técnicos, de desarrollo o de gestión y 3) pueden consultar bibliografía, documentación de estándares y ejemplos de documentos de otros proyectos.

- Auditorías del proceso y del producto. Los equipos de desarrollo han de pasar dos auditorías de su trabajo una hacia la mitad de la asignatura que cubre hasta el diseño de alto nivel y el proceso seguido y otra al final de la asignatura con el sistema y proceso completados. De la auditoría primera se produce un informe de auditoría que permite a los equipos de proyecto hacer una realimentación con comentarios procedentes de personal externo a su proyecto.

(Estamos bastante contentos con los resultados docentes que hemos obtenido con esta estrategia de proyecto.)

## **5. Papel de la programación y las herramientas dentro del curso**

El papel de la programación es importante en el sentido de que los alumnos están obligados a desarrollar un sistema de software completo que incluye toda la fase de codificación y test. Sin embargo, a nivel docente no nos preocupamos de las soluciones técnicas adoptadas (asumimos que ese aspecto ya ha sido cubierto docentamente), únicamente se exige y comprueba que el cada equipo de desarrollo dispone de un estándar de codificación y que éste ha sido seguido adecuadamente en la codificación del sistema. En alguna de las prácticas se exige también cierta codificación pero a nivel de la asignatura sólo nos interesa el buen funcionamiento de los programas pero no sus aspectos técnicos.

Por otra parte se aconseja (pero no se exige) el uso de herramientas por parte de los equipos para el desarrollo del proyecto, concretamente se aconseja el uso de herramientas de control de versiones, planificación de proyectos y de documentación de análisis y diseño y de desarrollo de aplicaciones (de las dos primeras se realizan prácticas tutoradas).

## **6. Lenguajes de programación utilizados en los trabajos prácticos**

En los trabajos de programación que han de realizar los alumnos nosotros aconsejamos y proporcionamos herramientas, documentación (libros, manuales y estándar de codificación), librerías (p.e. de comunicaciones) y soporte del lenguaje C++. Sin embargo damos libertad a los alumnos para que puedan realizar sus trabajos de codificación en cualquier otro lenguaje y entorno, siempre que cumplan los requisitos. Concretamente, además de en C++, distintos equipos han elegido Java, C, Visual Basic y Delphi como lenguajes de codificación.

## **7. Tipos de trabajos prácticos**

En cuanto a trabajos prácticos el presente curso hemos realizado 3 prácticas y el proyecto de curso:

- Práctica de control de versiones. Se utiliza la herramienta RCS de Unix para introducir al alumno en la problemática del control de versiones. La práctica es individual.
- Práctica de estimación y medida del tamaño del software. Por una parte se utiliza el método delphi para realizar la estimación del tamaño del software del proyecto de curso y luego se aplica COCOMO básico y COCOMO intermedio para tener una estimación del esfuerzo. Por otra parte, cada equipo de desarrollo tiene que realizar un programa para el cálculo automático del tamaño del software del proyecto del curso. Estos resultados serán utilizados para documentar el Plan de gestión del proyecto de software y el Documento Postmortem del proyecto. El trabajo se realiza por equipos de proyecto.

- Práctica de planificación de proyectos. El objetivo es adquirir experiencia en la planificación de un proyecto de software y en la utilización de herramientas que faciliten dicha planificación. Para ello el alumno parte una planificación ya realizada, que tiene ajustar en base a ciertas contingencias (alteración de los recursos del proyecto). Como herramienta en las prácticas se utiliza MS Project 4.0.
- Proyecto de curso. Es el trabajo de curso ya comentado anteriormente. En el presente curso consistía en un sistema para hacer que un PC corriendo windows pudiera funcionar de interfase de usuario para poder utilizar RCS que corre en una máquina HP con Unix remota.

## 8. Estrategias para incorporar la programación en el curriculum de la asignatura

De acuerdo a la filosofía antes comentada, consideramos muy importante que el proyecto de software que han de realizar los alumnos cubra también los aspectos de programación. Sólo pasando por esta etapa el alumno puede sensibilizarse realmente de toda la problemática que conlleva el desarrollo de un proyecto de software. En nuestro caso la programación ya está incorporada pero cara a la asignatura nos importan únicamente los aspectos de estándares y documentación.

## 9. Material didáctico utilizado

El material didáctico utilizado se puede clasificar en los siguientes apartados:

- Transparencias de clase y fotocopias que se proporcionan a los alumnos.
- Bibliografía que sirve de apoyo al curso, concretamente [MFMP94]. Los alumnos pueden utilizar también el libro [PRES93], del que ya disponen de las anteriores asignaturas de ingeniería del software. Además tienen acceso para consulta a una bibliografía más amplia de estándares.
- Ejemplos de documentos y planes encontrados en internet. Concretamente procedentes de cursos parcialmente “parecidos” de la Universidad de Carnegie Mellon.

## 10. Referencias

A continuación ponemos la bibliografía que hemos utilizado más extensamente para preparar la asignatura:

Textos básicos:

[MFMP94] C.Mazza, J.Fairclough, B.Melton, D. dePablo, A.Scheffer, R.Stevens. *Software Engineering Guides*, Prentice Hall. 1994.

Textos no básicos:

[HUMP95] W.S. Humphrey. *A Discipline for Software Engineering*. SEI Series in Software Engineering, Addison-Wesley. 1995. Caps. 4 y 5

[HUMP90] W.S. Humphrey. *Managing the Software Process*. SEI Series in Software Engineering, Addison-Wesley. 1990. Cap. 7.

[PRES93] R.S. Pressman. *Ingeniería del Software. Un enfoque práctico*. 3ª ed.. MacGraw Hill. 1993. Cap. 2.

[MFMP94b] C.Mazza, J.Fairclough, B.Melton, D. dePablo, A.Scheffer, R.Stevens. *Software Engineering Standards*. Prentice Hall. 1994.

Bibliografía de interés que sirvió para el planteamiento general del curso:

- [BBJS92] B. Bruegge, J. Blythe, J. Jacksin, J. Shufelt. *ObjectOriented System Modeling with OMT*. Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA '92), Vancouver, ACM Press, pp. 359-376, October 1992.
- [BRUE94] B. Bruegge. *From Toy System to Real System Development: Improvements in Software Engineering Education*. Workshop of the German Chapter of the ACM, B.G. Teubner Verlag, Stuttgart, pp. 62-72, February 1994.

## ÍNDICE

1. <i>Contexto de la asignatura</i> _____	1
2. <i>Objetivos del curso</i> _____	1
3. <i>Contenidos</i> _____	2
4. <i>Métodos de enseñanza utilizados</i> _____	3
5. <i>Papel de la programación y las herramientas dentro del curso</i> _____	4
6. <i>Lenguajes de programación utilizados en los trabajos prácticos</i> _____	4
7. <i>Tipos de trabajos prácticos</i> _____	4
8. <i>Estrategias para incorporar la programación en el curriculum de la asignatura</i> _____	5
9. <i>Material didáctico utilizado</i> _____	5
10. <i>Referencias</i> _____	5