La Ingeniería del Software en los planes de estudio: ¿dos perspectivas de una disciplina o más de lo mismo?

J.H. Canós, M.C. Penadés, V. Pelechano, J. Sánchez, I. Ramos, A. González
Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València

Camí de Vera, S/N, 46071 València
Tel: 3877350; Fax: 3877357;
e-mail: {jhcanos,mpenades,pele,jsanchez,iramos,agdelrio}@dsic.upv.es

Resumen

En este artículo se presenta la descripción del curriculum de Ingeniería del Software (IS) en la Universidad Politécnica de Valencia, presentando las asignaturas obligatorias relacionadas con la IS en las titulaciones de Ingeniero Técnico en Informática (ITI) e Ingeniero en Informática (II); nuestra intención, además de informar, es suscitar un debate en torno a una cuestión que a nuestro juicio es crucial para el diseño del conjunto de curricula en IS: ¿cuáles han de ser los conocimientos en IS que deben adquirir los Ingenieros Técnicos en Informática, por una parte, y los Ingenieros Superiores por otra? Nuestra propuesta consiste en diferentes temarios para las titulaciones medias y la superior, que han sido elaborados atendiendo a las que se supone deberían ser las competencias de los titulados en unas y otra pero que, desgraciadamente, el mercado muchas veces ignora. En el documento exponemos la motivación que nos ha llevado a elaborar los curricula en IS, presentando los contenidos de las asignaturas de IS en las titulaciones de ITIG e ITIS (Ingeniería del Software y Laboratorio de Ingeniería del Software), junto con los de la titulación de II (Ingeniería de Requisitos, Ingeniería de la Programación y Laboratorio de Ingeniería de la Programación).

Palabras clave: Docencia, Planes de Estudio, Ingeniería del Software.

1 Introducción

Los nuevos planes de estudio aprobados en las diferentes universidades españolas han supuesto un crecimiento espectacular de la presencia de la Ingeniería del Software (en adelante, IS) en los curricula de Informática, tanto a nivel medio como superior. Lo que antes se limitaba a una asignatura (o dos al máximo) en una carrera de cinco años ha pasado a ser una presencia constante a partir del segundo curso, a un nivel de troncalidad/obligatoriedad tal que esta disciplina es, sin duda, la más importante en los estudios: en la Universidad Politécnica de Valencia (UPV), por ejemplo, se cuenta actualmente con tres asignaturas obligatorias en la titulación de Ingeniero Técnico en Informática de Gestión (ITIG) y en la de Ingeniero en Informática (II), que son dos en la Ingeniería Técnica en Informática de Sistemas (ITIS), y algo parecido ocurre en la mayor parte de las Universidades del estado español.

Esto, que es sin duda una buena noticia, abre la posibilidad de diseñar con menos restricciones un curriculum apropiado para una disciplina tan amplia como la IS; en [3] se muestra la estructura de los estudios de IS en las titulaciones medias de la UPV. En este artículo vamos a completar la descripción del curriculum de IS en la UPV presentando, además, las asignaturas obligatorias relacionadas con la IS en la titulación de Ingeniero en Informática; nuestra intención, además de informar, es suscitar un debate en torno a una cuestión que a nuestro juicio es crucial para el diseño del conjunto de curricula en IS: *¿cuáles han de ser los conocimientos en IS que deben adquirir los Ingenieros Técnicos en Informática, por una parte, y los Ingenieros superiores por otra?*

El tema no es en absoluto trivial, y de hecho se tiende en general a repetir en las asignaturas de II gran parte de los contenidos estudiados en las asignaturas de ITIG e ITIS, lo cual nos parece de entrada incorrecto. Nuestra propuesta consiste precisamente en diferentes temarios para las titulaciones medias y la superior, que han sido elaborados atendiendo a las que se

supone deberían ser las competencias de los titulados en unas y otra pero que, desgraciadamente, el mercado muchas veces ignora.

Este trabajo está estructurado como sigue: en la sección 2 exponemos la motivación que nos ha llevado a elaborar los curricula que son presentados en las secciones posteriores. En la sección 3 se describen los contenidos de las asignaturas de IS en las titulaciones de ITIG e ITIS (Ingeniería del Software y Laboratorio de Ingeniería del Software), mientras que en la sección 4 nos dedicamos a las de la titulación de II (Ingeniería de la Programación, Laboratorio de Ingeniería de la Programación e Ingeniería de Requisitos). Finalmente, las conclusiones.

2 Motivación

La publicación en 1990 de las directrices para la elaboración de los Planes de Estudios en Informática supuso el punto de partida para la elaboración de los curricula de IS. Como es sabido, en dichas directrices se incluyen una serie de materias troncales que han de estar cubiertas en cualquier plan de estudios que se cree, dejando a las Universidades la posibilidad de incrementar los curricula con materias obligatorias y optativas. Un rápido vistazo a las materias troncales que con el nombre de Ingeniería del Software aparecen en los planes de estudio de ITIG, ITIS e II muestra gran similitud entre los contenidos de todas ellas. Esto muestra, de entrada, un claro concepto de qué es la IS como disciplina, y de su importancia en el proceso de construcción de aplicaciones.

Sin embargo, al mismo tiempo que se imponen una serie de contenidos, se permite que titulados medios puedan acceder a los estudios de II, lo que provoca que vuelvan a estudiar —de acuerdo al BOE— tópicos que ya han estudiado en el primer ciclo (lo cual es, desde luego, indeseable). Esta es, además, la situación en muchos centros del estado, donde los programas para las asignaturas de primer y segundo ciclo son muy similares (casi idénticos, en ocasiones). Es cierto que los planes de estudio de ITIG e ITIS, por un lado, y el de II por otro, no han sido diseñados precisamente pensando en ese paso de la titulación media a la superior, pero pensamos que en cualquier caso la materia es tan rica que los contenidos pueden diversificarse de manera plenamente justificada.

En efecto, basta con mirar, por un lado, a una noción tan básica de la IS como es el ciclo de vida de desarrollo de software, y por otro, a lo que en la bibliografía se recomienda como composición de los equipos de desarrollo de software (ver, por ejemplo, [10]), para darnos cuenta de cuáles son los conocimientos necesarios en cada caso. Tomando como ejemplo el ciclo de vida en cascada, parece evidente que ante fases tan diferenciadas y con objetivos tan claros se produzca una especialización de la gente en función de sus conocimientos. Así, un Ingeniero (superior) de Software debería centrar su trabajo (y, por tanto, sus conocimientos) en todo aquello relacionado con Planificación y Gestión de Proyectos, Garantía de Calidad, e incluso Análisis y Especificación de Requisitos, mientras que para un Ingeniero Técnico se reservarían las fases de Diseño, Codificación y Pruebas.

Esto no significa que éstos no deban conocer nada de Planificación ni de Análisis: efectivamente, deben ser capaces de entender un modelo de un sistema realizado con DFDs, o con diagramas OO, de la misma forma que deben ser capaces de entender un plan de proyecto para saber cuáles son sus restricciones de tiempo y recursos. Pero no tienen por qué estudiar a fondo técnicas de captura de requisitos, o técnicas de estimación para realizar un plan de proyecto. Lo mismo se puede afirmar de los Ingenieros superiores respecto a diseño de software, programación o técnicas de prueba, sólo que en este caso estaría más justificado un conocimiento amplio de estas disciplinas.

Con esta premisa, el curriculum de IS en la UPV se ha elaborado de la manera que se describe en las siguientes secciones. Se va a omitir, por razones de espacio, las materias optativas que componen la llamada intensificación en Ingeniería del Software del segundo ciclo de los estudios de II y todo lo relativo a las asignaturas de Proyectos Final de Carrera (ver detalles en [3]).

3 La IS en Ingeniería Técnica

El primer contacto con la IS en las carreras de ITIG e ITIS se produce en el quinto cuatrimestre cuando se estudia Ingeniería de Software, que tiene su continuación en Laboratorio de Ingeniería del Software en el sexto cuatrimestre. Seguidamente se dan detalles de sus objetivos y contenidos.

3.1 Ingeniería del Software (INS)

La docencia de la asignatura la imparte el Departamento de Sistemas Informáticos y Computación en los títulos de Ingeniero Técnico en Informática de Gestión (ITIG) e Ingeniero Técnico en Informática de Sistemas (ITIS), siendo troncal en ITIG y obligatoria en ITIS. La asignatura consta de 3 créditos teóricos y 3 prácticos.

Objetivos Generales

- Conocer la problemática del desarrollo de programas con calidad industrial, insistiendo en la necesidad de emplear técnicas de ingeniería.
- Estudiar los principios básicos de la ingeniería del software.
- Estudiar los paradigmas de ciclo de vida de desarrollo de software.
- Introducir las últimas tendencias en el desarrollo del software: El desarrollo de software orientado a objetos.

Programa

- **Software e Ingeniería del Software:** Se introduce una definición de software, así como sus características, su evolución en el tiempo y la aparición de la famosa "crisis del software". A continuación se resalta la importancia y la necesidad de utilizar una disciplina científica e ingenieril capaz de producir software de calidad.
- **Paradigmas del Ciclo de Vida:** Se presenta el concepto de ciclo de vida y se explican los paradigmas de ciclo de vida más conocidos: clásico o en cascada, clásico con prototipado, en espiral y el de la programación automática, haciendo hincapié en los pros y contras de cada uno de ellos.
- **Elementos de gestión de proyectos:** Se introducen los elementos necesarios para la gestión de un proyecto software. Se presentan los conceptos y las técnicas necesarias para la planificación temporal, el seguimiento y control del proyecto.
- **Desarrollo de software orientado a objetos:** Se plantea el paradigma orientado a objetos como una manera diferente de afrontar el desarrollo de software. Este tema es el más extenso de la asignatura y se divide en tres apartados:
  1. **Conceptos básicos de la orientación a objetos:** se presentan de forma genérica los conceptos esenciales de la orientación a objetos: clases, métodos, ejemplares, herencia, polimorfismo, enlace dinámico...
  2. **Introducción al Análisis y Diseño Orientado a Objetos:** se introduce al alumno en el análisis y diseño orientado a objetos mediante las técnicas: OMT y el diseño dirigido por responsabilidades.
  3. **Programación orientada a objetos:** En este tema se ilustran cada uno de los conceptos esenciales presentados con ejemplos en C++ y Object Pascal, insistiendo en las características comunes y en las diferencias a la hora de dar soporte a la orientación a objetos.
- **Técnicas de prueba de programas OO:** se presentan algunas técnicas de prueba no formal de programas aplicadas al caso de la programación orientada a objetos.

Prácticas de laboratorio

Las prácticas se realizan en grupos de dos personas a razón de dos horas semanales. A lo largo del curso se realizan las siguientes prácticas:

Práctica 1. Especificación de requisitos del software: se explica la guía de IEEE-830 para la especificación de requisitos software, introduciendo la especificación de aplicación como ejemplo práctico.

Práctica 2. Planificación, Control y Seguimiento de Proyectos de Software: se enseña el manejo de la aplicación MS Project 4.0 para la planificación, control y seguimiento de un proyecto software real.

Práctica 3. El lenguaje Object Pascal y su modelo de objetos: se presenta un curso práctico sobre el lenguaje Object Pascal y su modelo de objetos.

Práctica 4. El entorno de desarrollo Borland Delphi: Se presenta un curso básico sobre el manejo del Entorno Integrado de Desarrollo Borland Delphi, sin entrar en grandes detalles sobre los aspectos visuales del entorno. En este curso se pretende que el alumno utilice el Object Pascal puro, en modo consola y sin utilizar los componentes visuales que proporciona el entorno.

Práctica 5. Una práctica sencilla en Object Pascal: el alumno realizará una práctica sencilla en la que descubra los conceptos básicos de la programación orientada objetos sobre el lenguaje Object Pascal: creación y destrucción de ejemplares, especialización de clases, polimorfismo, enlace dinámico, clases abstractas...

Práctica 6. Práctica final: Una aplicación en Object Pascal: el alumno realizará una práctica final evaluable en la que pondrá en práctica todos los conocimientos adquiridos durante el curso (especificación de requisitos, planificación temporal, diseño y programación orientado a objetos) para implementar una aplicación real en Object Pascal que siga la filosofía orientada a objetos inculcada durante el curso.

3.2 Laboratorio de Ingeniería del Software (LIS)

La docencia de la asignatura la imparte el Departamento de Sistemas Informáticos y Computación en los títulos de Ingeniero Técnico en Informática de Gestión (ITIG) e Ingeniero Técnico en Informática de Sistemas (ITIS), siendo troncal en ITIG y optativa en ITIS. La asignatura se imparte en el segundo semestre del tercer curso de la Escuela Universitaria de Informática y consta de 3 créditos teóricos y 3 prácticos. Esta asignatura está concebida como la continuación aplicada de la asignatura de Ingeniería del Software, pensando siempre en que la mayoría de los alumnos que la cursan no van a continuar por el segundo ciclo.

Objetivos Generales

- Presentar los conceptos básicos de la programación en entornos visuales.
- Introducir conceptos avanzados de la programación orientada a objetos: construcción y utilización de componentes de software, reutilización...
- Introducir el modelo Cliente / Servidor y la repercusión del mismo en la construcción de aplicaciones.
- Introducir Internet cómo un nuevo ambiente de trabajo para el desarrollador actual y Java como su herramienta de desarrollo. Proponiendo las Intranets como el futuro inmediato.

Programa

- **Introducción al desarrollo de aplicaciones visuales.** El desarrollo rápido de aplicaciones y la Ingeniería del Software. Introducción a los entornos de programación visuales orientados a objetos. El entorno de desarrollo integrado *BORLAND DELPHI*, que incluye los temas referentes a la introducción a *Borland Delphi*, construcción de Proyectos con el IDE de Delphi, el diseño de la Interfaz de Usuario, construcción de aplicaciones (SDI y MDI), conceptos avanzados de Object Pascal y gestión de excepciones.

- **Desarrollo de aplicaciones de bases de datos y la nueva era cliente/servidor.** Desarrollo de aplicaciones de bases de datos en Delphi. Introducción al Borland Database Engine. BDE vs ODBC. Bases de datos locales y SQL. El control de la concurrencia y las transacciones. Introducción al modelo Cliente/Servidor. Construcción de aplicaciones distribuidas basadas en el modelo de servicios. El modelo de tres niveles. Construcción de aplicaciones multinivel con Delphi.

- **Conceptos Avanzados para el Desarrollo de Aplicaciones. Componentes y reutilización.** La biblioteca de componentes VCL. Desarrollo de componentes VCL y reutilización en Delphi. Un ejemplo práctico. Desarrollo con OLE/ActiveX. ¿Qué es la tecnología OLE/ActiveX?. COM un nuevo estándar de componentes. Automatización OLE y controles ActiveX.

- **Internet y el desarrollo de aplicaciones para el WEB:** Introducción a Internet y sus repercusiones en el desarrollo del software. El lenguaje Java. Pasado, presente y futuro. Desarrollo de aplicaciones para la WEB. El futuro inmediato.

Prácticas de laboratorio

Las prácticas se realizarán en el laboratorio en grupos de dos personas, utilizando como herramienta de desarrollo *Borland Delphi*. Los objetivos esenciales son: poner en práctica los conocimientos adquiridos en Ingeniería del Software (programación orientada a objetos y el análisis y diseño orientado a objetos) desde un punto de vista más aplicado y abordar: la programación en entornos visuales, la utilización y reutilización de componentes, aplicar algunos

conceptos del modelo Cliente/Servidor, para el acceso a bases de datos remotas. Durante el curso se realizarán las siguientes prácticas:

Práctica 1. Introducción a la programación en el IDE de Borland Delphi: tiene como objetivo familiarizar al alumno con el IDE de Delphi. Construyendo una sencilla aplicación de ventanas, para empezar a utilizar los componentes visuales que proporciona Delphi.

Práctica 2. Desarrollo de aplicaciones en Borland Delphi: engloba un conjunto de prácticas a realizar que consistirán en el desarrollo completo de aplicaciones:
1. Construcción de un editor de textos MDI.
2. Construcción de una aplicación de acceso a bases de datos.

Práctica 3. Creación de componentes en Borland Delphi:. En esta última práctica, se explicará y se pondrá en práctica la creación de nuevas componentes y se explicará cómo añadirlas a la biblioteca de componentes de *Delphi*.

4 La IS en Ingeniería Superior

El primer contacto con la IS en la titulación de II se produce en el séptimo cuatrimestre con las asignaturas de Ingeniería de Requisitos e Ingeniería de la Programación, que tienen su continuación en Laboratorio de Programación en el octavo cuatrimestre. Seguidamente se dan detalles de sus objetivos y contenidos.

4.1 Ingeniería de Requisitos (INR)

La docencia de la asignatura la imparte el Departamento de Sistemas Informáticos y Computación en la titulación de Ingeniero Informático, siendo una asignatura troncal que se imparte en el séptimo semestre. Consta de 4 créditos teóricos y 2 prácticos.

Objetivos Generales

El objetivo general de la asignatura es introducir a los alumnos en la disciplina de Ingeniería de Requisitos (INR). Este se desglosa en los siguientes:

- Dar una definición de la INR (conceptos básicos, estado actual y futuro de la IDR) y situarla dentro del proceso de desarrollo de software.
- Estudiar métodos de *análisis de requisitos software*, en especial el uso de métodos formales para *especificar requisitos software* en distintos marcos (marco lógico, marco algebraico y marco orientado a objetos).
- Introducir algunas nociones de prototipado y validación.

Programa

El contenido teórico de la asignatura se puede estructurar en 3 áreas temáticas o capítulos.

- ***Área temática 1. Conceptos Básicos.*** Esta parte pretende presentar una serie de factores que motiven el estudio de esta disciplina. Se dan una serie de definiciones de la IDR, haciendo hincapié en las actividades que conlleva. Se presentan las técnicas formales e intuitivas, así como los conceptos básicos. Finalmente se introducen las bases para la prototipación y validación. Esta área temática está estructurada en dos temas, denominados Introducción a la Ingeniería de Requisitos y Marco Conceptual de la Ingeniería de Requisitos.

- **Área temática 2.** ***Especificaciones formales en IDR.*** Una vez presentados los conceptos básicos, se entra de lleno en las especificaciones formales, estableciendo una comparativa frente a las no formales y viendo las ventajas e inconvenientes de unas y otras. A partir de aquí se entra de lleno en la modelización conceptual, la ejecutabilidad de las especificaciones y obtención de prototipos. Se aborda la modelización desde un marco lógico (tema denominado especificación lógica de requisitos) y un marco algebraico (tema denominado especificación algebraica de requisitos).

- **Área temática 3.** ***Especificaciones Orientadas a Objetos.*** En esta última parte del temario se introducen los fundamentos de la orientación a objetos, con la finalidad de poder llegar a modelar sistemas desde una perspectiva objetual. Una vez vista la modelización conceptual, al igual que en el área temática anterior se entra en aspectos de ejecutabilidad de la especificación y obtención de prototipos. Los contenidos anteriores están organizados en dos temas: el modelado orientado a objetos, donde se introduce a los

alumnos en la metodología OO-Method y la presentación de un lenguaje de especificación formal orientado a objetos, como puede ser el caso de L+0 ó OASIS.

Prácticas de laboratorio

Se realizan en grupos de 2 personas en un laboratorio docente. Las prácticas propuestas son:

- *Práctica 1. Especificación de Requisitos SW IEEE-830.* En esta práctica se presenta el estándar y los alumnos lo aplican a un caso práctico.
- *Práctica 2. Diagramas de Flujo de Datos.* El alumno realizará mediante una herramienta SW, los diagramas de flujo de datos de un ejemplo propuesto.
- *Práctica 3. Seminario de Prolog y Presentación de Visual Prolog o Sictus Prolog..* Se presentará el lenguaje Prolog y el entorno de trabajo para que el alumno se familiarice con él, resolviendo pequeños ejemplos.
- *Práctica 4. Modelización conceptual lógica (Aprox. operacional).* Se planteará un caso práctico para su resolución.
- *Práctica 5. Modelización conceptual lógica (Aprox. deductiva).* Se planteará un caso práctico para su resolución.
- *Práctica 6. Modelización conceptual orientada a objetos.* Se modelizará un caso práctico. Previamente se presenta al alumno la extensión del entorno para poder trabajar con objetos.

4.2 Ingeniería de la Programación (INP)

La asignatura Ingeniería de la Programación, troncal dentro de la titulación Ingeniero en Informática, se imparte en el séptimo semestre, con una carga docente de 4 créditos teóricos y 2 créditos de laboratorio.

Objetivos generales

- Estudiar la problemática del desarrollo de grandes sistemas, desde la etapa de análisis hasta la etapa de validación, resaltando la importancia de utilizar técnicas de ingeniería.
- Presentar y comparar diversas metodologías de desarrollo, desde el punto de vista de la orientación a objetos.
- Abordar la etapa de validación del sistema como una etapa mas del proceso de ingeniería.
- Introducir las últimas tendencias en el campo de los lenguajes visuales y de las herramientas CASE.

Programa

En la parte teórica se aborda la problemática del desarrollo de grandes sistemas, utilizando técnicas de ingeniería y desde el punto de vista de la orientación a objetos. Específicamente se abordan las etapas de análisis, diseño, implementación y validación.

La asignatura está dividida en tres grandes bloques: introducción a la ingeniería de la programación (10% de la materia), metodologías de desarrollo (70%) y técnicas de prueba de programas (20%).

Parte 1. Introducción a la ingeniería de la programación. En la introducción, se presentan los fundamentos de la ingeniería del software, junto a las nociones básicas de la terminología orientada a objetos, utilizando ejemplos en los lenguajes Simula, Eiffel y C++.

Parte 2. Metodologías de desarrollo. Este bloque constituye la parte fundamental del curso. Se estudia, con detenimiento, la metodología "Object Modeling Technique (OMT)", desde la etapa de análisis hasta la etapa de implementación. Se compara con el método unificado (UML) y el proceso de desarrollo, basado en los casos de uso, que subyace dentro de "Objectory".

Parte 3. Prueba no formal de programas. Se introducen las técnicas de prueba de programas, tanto estáticas (inspecciones de código), como dinámicas (caja blanca, caja negra) dentro del marco de la orientación a objetos.

Prácticas de laboratorio

Las prácticas se realizan en grupos de dos personas sobre PC compatibles. Están divididas en seis sesiones de 2 horas de duración cada una, organizadas de la siguiente forma:

- *Sesión 1.* Introducción a Visual C++, creación de aplicaciones utilizando AppWizard.
- *Sesión 2.* Inicio de la construcción de un editor gráfico elemental. Creación de clases base y derivadas.

- *Sesión 3*. Incorporación de las opciones de almacenamiento y recuperación de la información.
- *Sesión 4*. Toma de contacto con la herramienta CASE "System Architect". Utilización de las opciones relacionadas con OMT.
- *Sesión 5*. Modelizar distintos sistemas de información propuestos como ejercicio.
- *Sesión 6*. Presentación de la práctica final. Esta práctica entregada antes del examen de teoría. Debe documentarse utilizando la herramienta CASE e implementarse en Visual C++.

4.3 Laboratorio de Ingeniería de la Programación (LIP)

La asignatura "Laboratorio de Ingeniería de Programación" se imparte en el octavo semestre. Es una asignatura obligatoria con 6 créditos de carga. De estos 6 créditos 1,5 son de teoría y 4,5 de laboratorio.

Objetivos generales

El objetivo general de la asignatura es enfrentar al alumno con el desarrollo de un producto software.

- La asignatura pretende que el alumno adquiera una dimensión experimental para los conceptos y técnicas que ha ido aprendiendo a lo largo de la carrera.

Programa

La asignatura tiene mayor carga en prácticas de laboratorio, mientras que la carga teórica es mucho menor. Se trata, por tanto de una asignatura eminentemente práctica. El programa de la asignatura podría resumirse en el siguiente párrafo.

- Se parte de una especificación de requisitos de usuario suministrada por el profesor y el alumno debe generar una aplicación que satisfaga tales requisitos.

Prácticas de laboratorio

Se ha adoptado como herramienta de instrumentación *Power Builder 5.0*. Lo que permite abordar conceptos de orientación a objeto, bases de datos y diseño de interfaces.

Se propone un desarrollo cliente/servidor, pudiendo optar el alumno por diferentes servidores de datos.

La asignatura cuenta con 60 horas lectivas. Las 30 primeras se dedican al aprendizaje de la herramienta de desarrollo. Dicho aprendizaje se basan en sesiones tipo "tutorial" que el alumno puede seguir de forma flexible. En la segunda parte del curso los alumnos, organizados en grupos de tres, proceden al desarrollo de la aplicación según los requisitos establecidos.

Vinculación con otras asignaturas

Esta asignatura se relaciona con diferentes asignaturas del plan de estudios, pero muy especialmente con las que quedan dentro del ámbito de la Ingeniería del Software. Por lo que de forma directa se ha planteado un mecanismo de vinculación basado en la reutilización de casos prácticos.

El caso que se propone a los alumnos es presentado en otras asignaturas relacionadas. Concretamente este año se ha utilizado un caso de estudio especificado en la asignatura de Diseño de Bases de Datos de 3° e instrumentado en la asignatura de Gestión de Bases de Datos de 4° curso. Para el curso próximo se plantea la vinculación con IDP mediante el mismo mecanismo. La asignatura de laboratorio partiría de la especificación planteada como trabajo práctico para la asignatura de IDP para proceder a su desarrollo.

5 Conclusiones

En este artículo hemos mostrado cómo, desde un análisis del proceso de desarrollo de software y de las competencias laborales de Ingenieros técnicos y superiores, se puede llegar al diseño de un conjunto de curricula en IS para las diferentes titulaciones que evita coincidencias y que permite profundizar en cada caso en los aspectos más relevantes para el trabajo de los futuros titulados. Además de estas ventajas, permite dimensionar más adecuadamente cada una de las asignaturas, que permite ver partes de una disciplina tan amplia como la Ingeniería del Software.

Referencias

[1] Budd, T., Introducción a la programación orientada a objetos, Addison-Wesley Iberoamericana, 1994.

[2] Balzer R. et al. Software Technology in the 1990s: Using a New Paradigm IEEE Computer, Nov.1983

[3] Canós, JH.; Penadés, MC; Pelechano, V. *La Ingeniería del Software en los Planes de Estudio de la EUI de la UPV.* II Jornadas de Ingeniería del Software, San Sebastián, Septiembre, 1997.

[4] Davis, A.M, "201 Principles of Software Development" McGraw-Hill, Inc. 1994

[5] Jacobson I.,Christerson M.,Jonsson P.,Overgaard G., Object Oriented Software Engineering, a Use Case Driven Approach. Reading, Massachusetts. Addison-Wesley

[6] Goshling J., McGilton H. *The Java Language Environment. A White Paper.* Sun MicroSystems. 1995.

[7] Mynatt,B.,"Software Engineering with students Project Guidance", Prentice-Hall International,1990

[8] North, Ken ; Windows Multi-DBMS Programming : Using C++, Visual Basic, ODBC, OLE and Tools for DBMS Projects , John Wiley & Sons, 1995.

[9] Pacheco, X. ; Teixeira, S., Delphi Developer's Guide, Borland Press, Sams Publishing, 1995.

[10] Pressman, R. S., Ingeniería del Software: un enfoque práctico, 3ª ed., McGraw-Hill, 1994.

[11] Renaud, Paul E. ; Introduction to client/server systems : A practical guide for system professionals ; John Wiley, cop. 1993.

[12] Sayles, J.S. ; Karlen, S. ; Molchan, P. ; Bilodeau, G . , Gui_Based Design and Development for Client/Server Applications , John Wiley & Sons, 1994

[13] Todd, B. ; Kellen, Vince , Delphi : A Developer's Guide, M & T Books, 1995.

[14] Wirfs-Brock R.,Wilkerson B.,Wiener L., Designing Object Oriented Software. Englewood Cliffs NJ. Prentice- Hall