DATOS BÁSICOS DE LA GUÍA DOCENTE:

Materia:	INGENIERÍA DEL SOFTWARE		
Identificador:	30540		
Titulación:	GRADUADO EN INGENIERÍA INFORMÁTICA (SEMIPRESENCIAL). 2008 (BOE 15/12/2008)		
Módulo:	INGENIERÍA DEL SOFTWARE		
Tipo:	OBLIGATORIA		
Curso:	3	**Periodo lectivo:**	Primer Cuatrimestre
Créditos:	6	**Horas totales:**	150
Actividades Presenciales:	12	**Trabajo Autónomo:**	138
Idioma Principal:	Inglés	**Idioma Secundario:**	Castellano
Profesor:		**Correo electrónico:**	

PRESENTACIÓN:

Hoy en día uno de los pilares de la sociedad del bienestar reposa sobre complejos sistemas informáticos. Infraestructuras nacionales y diversidad de servicios dependen de ellos, y la mayor parte de los productos electrónicos incluyen una computadora y software de control. La fabricación industrial y la distribución están completamente informatizadas, como sucede con el sistema financiero. Por lo tanto, producir software de calidad y costeable es esencial para el funcionamiento de la economía nacional e internacional. La ingeniería del software es aquella disciplina de la ingeniería cuya meta es el desarrollo costeable y dentro de plazos de sistemas software que cumplan los requisitos establecidos de funcionalidad y calidad. La noción de ingeniería del software fue propuesta inicialmente en 1968 en una conferencia para discutir lo que en ese momento se llamó la “crisis del software”. A medida que los sistemas se volvían más grandes y complejos, los enfoques informales para el desarrollo de software se mostraron claramente insuficientes. Se necesitaban nuevas técnicas y métodos para controlar la complejidad inherente a estos sistemas grandes. Estas técnicas han llegado a ser parte de la ingeniería del software y son ampliamente utilizadas en la actualidad. No obstante, todavía hay muchos profesionales del software que no aplican de forma efectiva las técnicas de la ingeniería del software, con lo cual se sigue produciendo software que no satisface las necesidades del cliente, tanto en coste, calidad como funcionalidad. A pesar de que es ampliamente reconocido que no existe un enfoque “ideal” para la ingeniería del software, el conocimiento de sus técnicas, procesos y metodologías, los cuales se presentarán en esta asignatura, constituyen la mejor ayuda para que un futuro profesional de la informática pueda encarar con posibilidades de éxito sus retos laborales.

COMPETENCIAS PROFESIONALES A DESARROLLAR EN LA MATERIA:

Competencias Generales de la titulación	G01	Capacidad de liderazgo para poder influir sobre un colectivo con el fin de que este alcance unos determinados objetivos de forma conjunta y eficiente
	G02	Capacidad innovadora para proponer y encontrar formas nuevas y eficaces de realizar cualquier tarea y/ o función dentro de su entorno profesional con una elevada motivación por la calidad
	G03	Capacidad para trabajar dentro de equipos multidisciplinares para conseguir metas comunes, anteponiendo los intereses colectivos a los personales
	G04	Capacidad para trabajar siempre con responsabilidad y compromiso, creando un alto sentido del deber y el cumplimiento de las obligaciones
	G09	Capacidad para tomar decisiones de manera imparcial y desde un punto de vista racional
Competencias Específicas de la titulación	E07	Capacidad para trabajar eficazmente en equipos de proyecto, asumiendo en su caso responsabilidades directivas, y considerando los aspectos humanos, tecnológicos y financieros
	E08	Capacidad para comunicarse productivamente con clientes, usuarios y colegas, tanto de modo oral como por escrito, con el fin de transmitir ideas, resolver conflictos y alcanzar consensos
	E12	Capacidad para gestionar la complejidad a través de la abstracción, el modelado, las “best practices”, los patrones, los estándares y el uso de herramientas apropiadas
	E16	Capacidad para comprender un dominio de aplicación hasta el punto de ser capaz de desarrollar aplicaciones IT adecuadas para el mismo
	E17	Capacidad para identificar y analizar las necesidades de los usuarios con el objetivo de diseñar soluciones IT efectivas y usables que puedan integrarse en el entorno operativo del

		usuario.
	E18	Capacidad para identificar y definir los requisitos que deben ser satisfechos por los sistemas IT para satisfacer las necesidades planteadas por organizaciones o individuos
	E19	Capacidad para diseñar y definir la arquitectura de sistemas IT (software, hardware y comunicaciones) de acuerdo a unos requisitos consensuados entre las partes involucradas
	E20	Capacidad para realizar el diseño detallado de los componentes del proyecto (procedimientos, interfaz de usuario, características de equipos, parámetros de los sistemas de comunicaciones, etc.).
	E21	Capacidad para realizar pruebas que verifiquen la validez del proyecto (funcional, integridad de los datos, rendimiento de las aplicaciones informáticas, equipos, comunicaciones, etc.)
	E27	Capacidad para elaborar y mantener documentación descriptiva de la génesis, producción y operatividad de los sistemas informáticos
Resultados de Aprendizaje	R01	Analizar especificaciones de usuario.
	R02	Modelar dominios de problemas.
	R03	Sintetizar modelos de análisis del dominio de solución.
	R04	Diseñar soluciones a problemas.
	R05	Dominar la programación.
	R06	Dominar metodologías para la construcción organizada de software.
	R07	Comprender e interpretar documentos descriptivos de procesos software.
	R08	Interaccionar en inglés en un escenario de trabajo.
	R09	Realizar trabajos de investigación.
	R10	Trabajar productivamente en equipo.
	R11	Generar documentación técnica.

REQUISITOS PREVIOS:

Recomendable cursar esta asignatura junto con “Calidad de Software”.

PROGRAMACIÓN DE LA MATERIA:

Contenidos de la materia:

1 - Introduction to Software Engineering
1.1 - Introduction
1.2 - Motivation
2 - Modeling Techniques
2.1 - Analysis
2.2 - Design
2.3 - Case study
3 - Software Process
3.1 - Objectives
3.2 - The Software Process
3.3 - The Software Lifecycle
3.4 - Methodologies
3.5 - Case study
4 - Implementation and Software Testing
4.1 - Basic foundations
4.2 - Programming principles and guideliness
5 - Design and redesign
5.1 - Interface design patterns
5.2 - Design patterns
5.3 - Refactorings

La planificación de la asignatura podrá verse modificada por motivos imprevistos (rendimiento del grupo, disponibilidad de recursos, modificaciones en el calendario académico, etc.) y por tanto no deberá considerarse como definitiva y cerrada.

METODOLOGÍAS Y ACTIVIDADES DE ENSEÑANZA Y APRENDIZAJE:

Metodologías de enseñanza-aprendizaje a desarrollar:

- Sesiones teóricas, casos prácticos y resolución de ejercicios. Clases magistrales de transmisión deconocimientos por parte del profesor, con participación activa de los estudiantes.
- Sesiones de tutoría. Las sesiones de tutoría se realizarán bajo demanda. Se fomentará en estas sesiones el uso de tecnologías no presenciales.
- Trabajo autónomo. Los alumnos deberán estudiar el material presentado, tratar de resolver losejercicios y problemas propuestos.
- Trabajo en equipo.Los alumnos deberán trabajar productivamente en equipo.

Volumen de trabajo del alumno:

Modalidad organizativa	Métodos de enseñanza	Horas estimadas
Actividades Presenciales	Clase magistral	4
	Otras actividades teóricas	0,5
	Casos prácticos	1
	Resolución de prácticas, problemas, ejercicios etc.	1,5
	Debates	1
	Exposiciones de trabajos de los alumnos	1
	Actividades de evaluación	3
Trabajo Autónomo	Asistencia a tutorías	8
	Estudio individual	50
	Preparación de trabajos individuales	20,5
	Preparación de trabajos en equipo	23
	Tareas de investigación y búsqueda de información	7,5
	Lecturas obligatorias	10
	Lectura libre	9
	Otras actividades de trabajo autónomo	10
	Horas totales:	150

SISTEMA DE EVALUACIÓN:

Obtención de la nota final:

Pruebas escritas:	60	%
Trabajos individuales:	15	%
Trabajos en equipo:	20	%
Participación:	5	%
TOTAL	100	%

*Las observaciones específicas sobre el sistema de evaluación serán comunicadas por escrito a los alumnos al inicio de la materia.

BIBLIOGRAFÍA Y DOCUMENTACIÓN:

Bibliografía básica:

Pressman, Roger (2005). Software Engineering. A Practitioners Approach, 6th Ed., McGraw-Hill.
Sommerville, Ian (2004). Software Engineering, 7th Ed., Pearson.

Bibliografía recomendada:

Peckham, Joan (ed.) (2003). Practicing Software Engineering in the 21st Century, IRM Press.
Bjørner, Dines (2006). Software Engineering 3. Domains, Requirements and Software Design, Springer.
Shoval, Peretz (2007). Functional and Object Oriented Analysis and Design: an Integrated Methodology, Idea Group Publishing.

Jalote, Pankaj (2005). An Integrated Approach to Software Engineering, Springer.
McConnell, Steve (2003). Professional Software Development, Addison Wesley
Kroll, Per (2006). Agility and Discipline Made Easy: Practices from OpenUP and RUP, Addison Wesley.
Tomayko, James et al (2004). Human Aspects of Software Development, Charles River Media.
Sangwan, Raghvinder et al (2007). Global Software Development Handbook, Auerbach Publications.
Aurum, Aybüke et al (2005). Engineering and Managing Software Requirements, Springer.
Gunderloy, Mike (2004). Coder to Developer: Tools and Strategies for Delivering Your Software, Sybex
Booch, Grady et al (2007). Object-Oriented Analysis and Design with Applications, 3th Ed., Addison-Wesley.
Pidd, Michael (ed.) (2004). Systems Modelling. Theory and Practice, John Wiley.

Páginas web recomendadas:

Center for Systems and Software Engineering: The aim of this site is to work towards evolving and unifying theories and practices of systems and software Engineering.	http://csse.usc.edu/csse/
IEEE Transactions on Software Engineering: Technical articles and news about Software Engineering issues	http://www.computer.org/portal/site/transactions
The Podcast for Professional Software Developers: Here you can download audio episodes relating experiences of software engineers	http://www.se-radio.net/

* Guía Docente sujeta a modificaciones