GUÍA DOCENTE

INGENIERÍA DEL SOFTWARE

Coordinación: Juan Manuel Gimeno

Año académico 2014-15

INGENIERÍA DE SOFTWARE 2014-15

Información general de la asignatura

Denominación	INGENIERÍA DEL SOFTWARE
Código	102018
Semestre de impartición	1r Q Evaluación Continuada
Carácter	Obligatoria
Número de créditos ECTS	6
Créditos teóricos	3
Créditos prácticos	3
Coordinación	Juan Manuel Gimeno
Horario de tutoría/lugar	Juan Manuel Gimeno (3.20 EPS, miércoles a las 13h; otras con cita previa) Montserrat Sendín (3.20 EPS, con cita previa)
Departamento/s	Informàtica i Enginyeria Industrial
Distribución carga docente entre la clase presencial y el trabajo autónomo del estudiante	40% presencial 60% trabajo autónomo
Modalidad	Presencial
Información importante sobre tratamiento de datos	Consulte este enlace para obtener más información.
Idioma/es de impartición	Catalán 60% Castellano 30% Inglés 10%
Grado/Máster	Grado en Ingeniería Informática
Distribución de créditos	Juan Manuel Gimeno Illa 4.5 Montserrat Sendin Veloso 4.5
Horario de tutoría/lugar	Juan Manuel Gimeno (3.20 EPS, miércoles a las 13h; otras con cita previa) Montserrat Sendín (3.20 EPS, con cita previa)
Dirección electrónica profesor/a (es/as)	jmgimeno@diei.udl.cat msendin@diei.udl.cat

INGENIERÍA DE SOFTWARE 2014-15

Juan Manuel Gimeno Illa
Montserrat Sendin Veloso

Información complementaria de la asignatura

Recomendaciones

Para cualquier duda y / o cuestión se recomienda enviar un correo electrónico al profesorado de la asignatura. Resolver los problemas y ejercicios de programación que se proponen diariamente permite alcanzar los objetivos de aprendizaje establecidos.

Asignatura/materia en el conjunto del plan de estudios

Asignatura que se imparte durante el 2 º semestre del 2 º curso de la titulación Grado en Ingeniería Informática. Corresponde a la Materia "Informática" dentro del Módulo de "Formación Básica".

Objetivos académicos de la asignatura

- Comprender el concepto de código como algo que evoluciona en el tiempo.
- Reconocer el concepto de responsabilidad como fundamental para plantear un diseño orientado a objetos.
- Conocer las bases conceptuales y los diferentes aspectos de la disciplina.

- Aplicar la técnica de los casos de uso.
- Desarrollar el diagrama de clases del análisis siguiendo los principios del Modelado Orientado a Objetos.
- Elaborar los contractos de las operaciones.
- Usar una herramienta de modelado basada en UML
- Conocer los diferentes modelos de proceso del ciclo de vida del software usados a lo largo de los años.
- Comprender la filosofía de desarrollo utilizada en el Proceso Unificado
- Ser capaz de programar pruebas unitarias básicas.
- Comprender y saber aplicar los principios fundamentales del diseño orientado a objetos.

- Especificar textualmente las necesidades funcionales de un determinado sistema software planteado a través de un enunciado u otras informaciones procedentes del usuario.
- Especificar textualmente los requisitos no funcionales de un determinado sistema software planteado a traves de un enunciado u otras informaciones procedentes del usuario.
- Expresar gráficamente el flujo de eventos, como conjunto de entradas y salidas, que describe el comportamiento del sistema.

Competencias

- **Competencias transversales**
  - Capacidad de comprender las necesidades del usuario expresadas en un lenguaje no técnico.
- **Competències específicas**
  - Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social.
  - Conocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos.
  - Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas de información, incluidos los basados en web.
  - Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.
  - Capacidad para diseñar y evaluar interfaces persona computador que garanticen la accesibilidad y

usabilidad a los sistemas, servicios y aplicaciones informáticas.

Contenidos fundamentales de la asignatura

Tema I - *Aspectos introductorios*

- Cuestiones iniciales sobre la Ingeniería del Software

- Un poco de historia

- Proceso de desarrollo de software

- Modelos de proceso de software

- Conclusiones

Tema II - *Especificaciones y Requisitos*

- Análisis de requisitos

  - Conceptos iniciales
  - Tipos de requisitos
  - Ejemplos

- La técnica de los Casos de Uso

  - Conceptos y componentes
  - Identificación y especificación de los casos de uso
  - Ejemplos

- Un paso más en la especificaciόm: Los Diagramas de Secuencia del Sistema

- Conclusiones

Tema III - *Análisis del Dominio*

- Diagramas de Clases del Análisis

  - Concepción
  - Fundamentos del Modelado Orientado a Objetos
  - Construcción del Modelo del dominio
  - Ejemplos

- Un paso más en el análisis: los Contratos de las operaciones

- Conclusiones

Tema IV - Introducción al diseño y pruebas unitarias

- El código como algo que varía en el tiempo.

  - El concepto de degradación del código
  - Síntomas de un codi degradado

- El framework JUnit para pruebas unitarias

- Las pruebas como especificaciones ejecutables
- Las pruebas como facilitadores de cambio
- Código testable y código flexible

Tema V - *Los principios SOLID*

- Principio de responsabilitat única
- Principio abierto-cerrado
- Principio de substitución de Liskov
- Principio de segregación de interficies
- Principio de inversión de dependencias

Tema VI - *Diseño basado en responsabilidades*

- El concepto de responsabilidad
- Patrones GRASP de assignación de responsabilidades

Ejes metodológicos de la asignatura

PARTE PRESENCIAL

Grupos Grandes

• **Clases Teoría (3 créditos)**

- Clases soportadas con transparencias y/o apuntes
- Se trabaja siempre con ejemplos

Grupos Medianos

• **Clases Laboratorio (3 créditos)**

- Uso de la herramienta de Modelado en UML: Visual Paradigm
- Trabajo continuado alrededor de un cierto **enunciado de práctica**, que simulará el desarrollo de un proyecto de software

PARTE NO PRESENCIAL

- La práctica se completará en horas **No Presenciales**
- **Se recomienda** que el alumno resuelva por cuenta propia los problemas de la **colección de problemas**, con objeto de obtener feedback por parte del profesor

Plan de desarrollo de la asignatura

Semana	Contenido
1	Presentación+Inicio T-I (hasta Crisi SW) T-I (Procesos desarrollo)
2	T-I- (Modelos Proceso + PU) Inicio T-II

3	Fiesta T-II-1ªparte
4	T-II- 2ª parte T-II – 2ª parte - Prob.
5	Dg. CU – Prca. T-III
6	Dg. CU Prca.Correc. + Rq. F y NF T-III
7	Fiesta Dg Clas. con herramienta + tiempo para Prca.
8	T-III (Ejemplos Dg. Clas. + Contratos) Contratos
9	Primer parcial
10	T-IV Introducción al diseño T-IV Pruebas unitarias - Conceptos
11	T-IV Pruebas unitarias - Junit T-IV Pruebas unitarias - Junit
12	T-IV Pruebas unitarias - Fexibilidad T-IV Pruebas unitarias - Fexibilidad
13	T-V Principios SOLID - Introducción y SRP T-V Principios SOLID - OCP y LSP
14	T-V Principios SOLID - ISP y DIP T-VI: Diseño basado en responsabilidades - Conceptos
	Navidad
15	T-VI: Diseño basado en responsabilidades - Patrones GRASP T-VI: Diseño basado en responsabilidades - Patrones GRASP
16	Segundo parcial
17	
18	Tutorias
19	Recuperaciones

Sistema de evaluación

Evaluación continuada

• **50% Teoria**

- **Parcial 1**: 25%

- **Parcial 2**: 25%
- Si Parcial 1 ó Parcial 2 < 4 = Recuperación de la/as parte/s correspondiente/s
- Tipología de examen: fijación de conceptos y resolución de problemas

• **50% Práctica**

- Trabajo en parejas (Análisis) / individualmente (Diseño)
- Entregas programadas, fechas no prorrogables
    - Análisis:
        - Análisis de Requisitos: 15%
            - Pre-entrega: Diagrama de Casos de Uso
        - Modelo del Dominio: 15%
    - Diseño:
        - Pruebas unitarias: 20%

• **Requisitos:**

- **Mínimo de 4** en cada parte teórica para ponderar con la nota de prácticas
- **Aprovado** = Nota Final >= 5

Bibliografía y recursos de información

BIBLIOGRAFIA BASICA

- Temas introductorios:

- I. Sommerville: “Ingeniería de Software”.
  Prentice Hall, 2005 (7ª ed.)

- Requerimientos:

- G. Kotonya, I. Sommerville: “Requirements Engineering: Processes and Techniques”.
  Wiley, 1998
- A. Sutcliffe: “User-Centred Requirements Engineering. Theory and Practice”.

  Springer-Verlag, 2002

- Pruebas unitarias:

- P.Tahchiev et al.: Junit in Action (2nd edition). Manning, 2011.

- Metodología orientada a objetos:

- C. Larman: “Applying UML and Patterns: An Introduction to Object- Oriented Analysis and Design and Iterative Development”
  Prentice-Hall, 2005 (3ª ed.)
  Versión española: “UML y Patrones”_ Prentice-Hall, 2002 (2ª ed.)
- Robert C. Martin: "Agile Software Development: Principles, Patterns, and Practices", Prentice-Hall, 2002.
- G. Booch, J. Rumbaugh, I. Jacobson: _“El Lenguaje Unificado de Modelado”._

Addison-Wesley, 2006 (2ª ed.)

- J. Rumbaugh, I. Jacobson, G. Booch: "El Lenguaje Unificado de Modelado. Manual de referencia"._

  Addison-Wesley, 2000

- I. Jacobson, G. Booch, J. Rumbaugh: _ "El Proceso Unificado de Desarrollo de Software".
  Addison-Wesley, 2000

BIBLIOGRAFIA COMPLEMENTARIA

- Temas introductorios:

- R. S. Pressman: "Ingeniería de Software: Un enfoque práctico".

  McGraw-Hill, 2005 (6ª ed.)

- Requerimientos:

- D. Kulak, E. Guiney: "Use Cases, Requirements in Context".

  Addison Wesley, 2000

- I. Jacobson: "Object-Oriented Software Engineering. A Use Case Driven Approach".

  Addison-Wesley, 1992

- Metodología orientada a objetos:

- M. Fowler, K. Scout: "UML Gota a Gota".

  Addison-Wesley, 1999

- M. Fowler: "Refactoring: Improving the Design of Existing Code", Addison-Wesley, 1999.
- J. Conallen: "Building Web Applications with UML". Addison Wesley, 1999