GUÍA DOCENTE

INGENIERÍA DEL SOFTWARE

Año académico 2013-14

INGENIERÍA DE SOFTWARE 2013-14

Información general de la asignatura

Denominación	INGENIERÍA DEL SOFTWARE
Código	102018
Semestre de impartición	1r Q Avaluació Continuada
Carácter	Obligatòria
Número de créditos ECTS	6
Créditos teóricos	0
Créditos prácticos	0
Departamento/s	Informàtica i Enginyeria Industrial
Información importante sobre tratamiento de datos	Consulte este enlace para obtener más información.
Idioma/es de impartición	Català 60% Castellà 30% Anglès 10%
Distribución de créditos	Juan Manuel Gimeno Illa 4.2 Montserrat Sendin Veloso 4.2

INGENIERÍA DE SOFTWARE 2013-14

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

Ver apartado de competencias.

Competencias

Competencias específicas de la titulación

- 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.

  Objetivos

  - Comprende 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.

- Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.

  Objetivos

  - 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.

Competencias transversales de la titulación

- Capacidad de comprender las necesidades del usuario expresadas en un lenguaje no técnico.

  Objetivos

- 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.

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

PART PRESENCIAL

Grups Grans

• **Classes Teoria (3 crèdits)**

- Classes suportades amb transparències i/o apunts
- Es treballa sempre amb exemples

Grups Mitjans

• **Classes Laboratori (3 crèdits)**

- Us de l'eina de Modelat en UML: Visual Paradigm
- Treball continuat al voltant d'un cert **enunciat de pràctica**, que simularà el desenvolupament d'un projecte de software

PART NO PRESENCIAL

- La pràctica es completarà en hores **No Presencials**
- **Es recomana** que l'alumne resolgui per compte propi els problemes de la **col·lecció de problemes**, a fi d'obtenir feedback per part del professor

Sistema de evaluación

Evaluación continuada

• **50% Teoria**

- **Parcial 1**: 25%

- **Parcial 2**: 25%
- Si Parcial 1 ó Parcial 2 < 4 = Recuperació de la/es part/s corresponent/s
- Tipologia d'examen: fixació de conceptes i resolució de problemes

• **50% Pràctica**

- Treball en parelles (Anàlisi) / individualment (Disseny)
- Entregues programades, dates no prorrogables
    - Anàlisi:
        - Anàlisi de Requeriments: 15%
            - Pre-entrega: Diagrama de Casos d'Ús
        - Model del Domini: 15%
    - Disseny:
        - Proves unitàries: 20%

• **Requisits:**

- **Mínim de 4** en cada part teòrica per ponderar amb la nota de pràctiques
- **Aprovat** = 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