Desarrollo de Software

Curso Académico: (2024 / 2025) **Fecha de revisión: 26-04-2024**

Departamento asignado a la asignatura: Departamento de Informática

Coordinador/a: LOPEZ CUADRADO, JOSE LUIS

Tipo: Obligatoria Créditos ECTS : 6.0

Curso : 2 Cuatrimestre : 2

REQUISITOS (ASIGNATURAS O MATERIAS CUYO CONOCIMIENTO SE PRESUPONE)

Programación (Curso 1 / Cuatrimestre: 1)
Estrutura de datos y Algoritmos (Curso: 1 / Cuatrimestre 2)
Ingeniería del Software (Curso 2 / Cuatrimestre 1)
Teoría de Autómatas y Lenguajes Formales (Curso 2 / Cuatrimestre 1)

COMPETENCIAS Y RESULTADOS DEL APRENDIZAJE

¿ Concebir, diseñar y aplicar técnicas de pruebas funcionales, estructurales y unitarias que sirvan para la verificación de software en distintos procesos de desarrollo de software
¿ Conocer y adquirir conciencia de los fundamentos éticos y legales de la profesión de la Ingeniería Informática.

OBJETIVOS

El objetivo del curso es permitir al estudiante profundizar en prácticas ágiles de desarrollo que facilitan la obtención de componentes software con alta calidad. Se aprenderán técnicas de pruebas funcionales y estructurales, y se aplicarán mediante pruebas unitarias automatizadas en un proceso de Desarrollo Dirigido por Pruebas. Así mismo se tratarán los fundamentos éticos y legales de la profesión de la Ingeniería Informática. Finalmente se aprenderán técnicas de refactorización de código y se aplicarán conceptos de diseño simple y patrones de diseño para la asignación de responsabilidades.

DESCRIPCIÓN DE CONTENIDOS: PROGRAMA

1.- Prácticas genéricas del desarrollo ágil
1.1.- El proceso de desarrollo de software.
1.2.- Programación en Parejas
1.3.- Estándares de codificación
1.4.- Propiedad Colectiva de Código
2.- Desarrollo Dirigido por Pruebas
2.1.- Principios del Desarrollo Dirigido por Pruebas
2.2.- Técnicas de Prueba Funcionales
2.3.- Técnicas de Prueba Estructurales
2.4.- Automatización de Pruebas Unitarias
2.5.- Integración Continua Automatizada
3.- Refactoring y Diseño Simple
3.1.- Refactoring
3.2.- Principios de Diseño Simple
3.3.- Patrones de Diseño para la Asignación de Responsabilidades
4.- Aspecto éticos y legales en la profesión de Ingeniero de Software.
4.1.- La profesión de Ingeniero de Software
4.2.- Código ético de la profesión de Ingeniero de Software.

ACTIVIDADES FORMATIVAS, METODOLOGÍA A UTILIZAR Y RÉGIMEN DE TUTORÍAS

- Clases Teóricas: 1,5 ECTS. Tienen por objetivo alcanzar las competencias específicas cognitivas y procedimentales de la asignatura

- Clases Prácticas: 1,5 ECTS. Desarrollan las competencias específicas instrumentales y la mayor parte de las transversales,s. También tienen por objetivo desarrollar las capacidades específicas actitudinales. Consisten en el diseño y desarrollo de un componente de software. En el ámbito de esta actividad de desarrollo se ejercitarán las técnicas de especificación de requisitos, diseño, revisión de calidad y pruebas del código que implementa el componente. Estas actividades se realizarán en equipo. Todo ello desarrollado conforme a los aspectos éticos y legales de desarrollo del proceso.

- Trabajos Prácticos 2 ECTS: elaborados con y sin presencia del profesor sobre un caso de proyecto de desarrollo de un componente de software, donde se profundice sobre todos los aspectos teóricos abordados en la materia aspecto de la materia.

- Tutorias: Asistencia individualizada (tutorías individuales) o en grupo (tutorías colectivas) a los estudiantes por parte del profesor.

- Examen Final: 1 ECTS. Tiene por objeto incidir y complementar en el desarrollo de las capacidades específicas cognitivas y procedimentales.

SISTEMA DE EVALUACIÓN

Peso porcentual del Examen Final:	40
Peso porcentual del resto de la evaluación:	60

Los ejercicios y exámenes además de servir como actividad formativa tienen el doble objetivo de ser medida para el sistema de evaluación. El sistema de evaluación incluye la valoración de las actividades académicas dirigidas y prácticas según la siguiente ponderación.

Ejercicios Guiados (Evaluación Continua): 30%
Exámenes intermedios (Evaluación Continua): 30%
Examen Final: 40%

Para superar la asignatura, es imprescindible aprobar, por separado, cada una de las partes (Exámenes intermedios, Ejercicios Guiados y Examen). Cada parte puede promediar a partir de 4 sobre 10.

BIBLIOGRAFÍA BÁSICA

- Beck, Ken, et al.. Test-Driven Development By Example. , Three Rivers Institute., 2002

- Beck, Ken. Una explicación de la Programación Extrema, Addison-Wesley, 2000

- Craig S. Larman Applying UML and Patterns., Pearson Education . 3er Edition, 2012

- Fowler, Martin et al.. Refactoring: Improving the Design of Existing Code. , Addison-Wesley. , 1999

- Lee Copeland. A Practitioner's Guide to Software Test Design., Artech House Publishers, 2003

BIBLIOGRAFÍA COMPLEMENTARIA

- Paul C. Jorgensen Software Testing: a craftsman's approach. , CRC.

- Roger S. Pressman. Ingeniería del Software. Un enfoque práctico. , McGraw Hill. 7ª Edición., 2009

- Spyros Xanthakis, Michel Maurice, Antonio de Amescua, Olivier Houri, Luc Griffet. Test and contro¿le des logiciels : me¿thodes, techniques and outils, EC2..