Desarrollo de Software

Curso Académico: (2019 / 2020) **Fecha de revisión: 06-05-2020**

Departamento asignado a la asignatura: Departamento de Informática

Coordinador/a: GARCIA GUZMAN, JAVIER

Tipo: Obligatoria Créditos ECTS : 6.0

Curso : 3 Cuatrimestre : 2

REQUISITOS (ASIGNATURAS O MATERIAS CUYO CONOCIMIENTO SE PRESUPONE)

Programación
Ingeniería de Software

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.

Los resultados del aprendizaje serán los siguientes:

R1 (Conocimiento y comprensión: Tener conocimientos básicos y la compresión de los fundamentos científicos y tecnológicos de la Ingeniería Informática, así como un conocimiento específico de las ciencias de la computación, la ingeniería de computadores y sistemas de información).,
R2 (Análisis de la Ingeniería: Ser capaces de identificar problemas de Ingeniería Informática, reconocer sus especificaciones, establecer diferentes métodos de resolución y seleccionar el más adecuado para su solución, teniendo en cuenta las en limitaciones sociales, salud humana, Medio Ambiente, y comerciales aplicables en cada caso).,
R3 (Diseño en Ingeniería: Ser capaces de realizar diseños de ingeniería de acuerdo a su nivel de conocimiento y comprensión que cumplan con las especificaciones requeridas colaborando con otros ingenieros y titulados. El diseño abarca dispositivos, procesos, métodos y objetos, y especificaciones más amplias que las estrictamente técnicas, lo cual incluye conciencia social, salud y seguridad, y consideraciones medioambientales y comerciales).

1. Las Competencias transversales o genéricas que se adquirirán a lo largo de la asignatura son:
1.1 Capacidad de abstracción
1.2 Capacidad de análisis y síntesis (PO: b)
1.3 Capacidad de organizar y planificar (PO: b)
1.4 Capacidad para resolver problemas (PO: a, e, k)
1.5 Trabajo en equipo (PO: d)
1.6 Capacidad para aplicar los conocimientos en la práctica (PO: a, b, d, e, g, k)

2. Las competencias especificas que se desarrollarán a lo largo del curso son:
2.1 Cognitivas (Saber) (PO: a, b, d, e, f, g, k) (CECRI8)
2.1.a Conocimiento sobre modelos de ciclos de vida y proceso de desarrollo de software basados en el desarrollo dirigido por pruebas (CECRI16)
2.1.b. Cultura de desarrollo ágil
2.1.c. Técnicas facilitadoras del desarrollo ágil
2.1.d. Desarrollo Dirigido por las pruebas
2.1.e. Diseño Simple y Refactoring
2.1.f Conceptos éticos en el proceso de desarrollo de software (CECRI1)
2.2 Procedimentales/Instrumentales (Saber hacer) (PO :b, e, k)
2.2.a. Desarrollar un componente software aplicando normativas de código
2.2.b. Desarrollar un componente software aplicando propiedad colectiva de código
2.2.c. Desarrollar un componente software aplicando desarrollo dirigido por pruebas
2.2.d. Desarrollar un componente software aplicando técnicas de prueba estructurales y funcionales
2.2.e. Desarrollar un componente software aplicando mecanismos de prueba automatizada
2.2.f. Desarrollar un componente software aplicando mecanismos de integración continua y automatizada
2.2.g. Desarrollar un componente software aplicando refactoring
2.2.h. Desarrollar un componente software aplicando patrones de diseño

2.3 Actitudinales (Ser) (PO: e, k)
2.3.a Capacidad para generar nuevas ideas
2.3.b Preocupación por la calidad en el desarrollo de un proyecto
2.3.c Motivación de logro ante problemas
2.3.d Interés por investigar y buscar soluciones a nuevos problemas
2.3.e Preocupación por el desarrollo ético de un proyecto

DESCRIPCIÓN DE CONTENIDOS: PROGRAMA

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

ACTIVIDADES FORMATIVAS, METODOLOGÍA A UTILIZAR Y RÉGIMEN DE TUTORÍAS

- Clases Teóricas: 1,5 ECTS. (PO: a, b, e, f, g, k)
Tienen por objetivo alcanzar las competencias específicas cognitivas y procedimentales de la asignatura

- Clases Prácticas: 1,5 ECTS. (PO: a, b, d, e, f, g, k)
Desarrollan las competencias específicas instrumentales y la mayor parte de las transversales, como son la de trabajo en equipo, capacidad de aplicar los conocimientos a la práctica, de planificar y organizar y de análisis y síntesis. 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.

- Realización de Actividades Académicas Dirigidas
a) Con presencia del profesor: 1 ECTS Planteamiento de un trabajo, 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. (PO: a, b, d, e, f, g, k)
b) Sin presencia del profesor: 1 ECTS. Desarrollo y finalización del trabajo relacionado con el desarrollo de un componente de software aplicando todos los aspectos teóricos abordados en la materia aspecto de la materia. (PO: a, b, e, f, k)

- Examen: 1 ECTS. Tiene por objeto incidir y complementar en el desarrollo de las capacidades específicas cognitivas y procedimentales. (PO: a, b, d, e, f, g, k)

SISTEMA DE EVALUACIÓN

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: 30% (PO: a, b, d, e, f, g, k)

Tests de Teoría: 30% (PO: a, b, d, e, f, g, k)

Práctica Final (Examen): 40% (PO: a, b, d, e, f, g, k)

Para superar la asignatura, es imprescindible aprobar, por separado, cada una de las partes (Teoría, Ejercicios Guiados y Práctica Final).

Es imprescindible superar, por separado, cada una de las partes de la asignatura para poder superarla

a nivel global.

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

BIBLIOGRAFÍA BÁSICA

- Beck, Ken Una explicación de la Programación Extrema, Addison-Wesley, 2000
- Beck, Ken, et al. Test-Driven Development By Example, Three Rivers Institute, 2002
- 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 .

BIBLIOGRAFÍA COMPLEMENTARIA

- Paul C Jorgensen Software Testing: a craftsman's approach, CRC Press.
- 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 & Controle des Logiciels: Methodes, Techniques & Outils, EC2.