CENTRO UNIVERSITARIO DE TECNOLOGÍA Y ARTE DIGITAL

PLANIFICACIÓN DE LA DOCENCIA UNIVERSITARIA

GUÍA DOCENTE

Diseño de Software

1. DATOS DE IDENTIFICACIÓN DE LA ASIGNATURA.

Título:	Grado en Ingeniería del Software
Facultad:	Centro Universitario de Tecnología y Arte Digital (U-TAD)
Materia:	Ingeniería del Software
Denominación de la asignatura:	Diseño de Software
Curso:	2
Cuatrimestre:	2
Carácter:	Obligatoria
Créditos ECTS:	6
Modalidad/es de enseñanza:	Presencial
Idioma:	Castellano
Profesor/a:	Ramona Ruiz Blázquez
E-mail:	ramona.ruiz@u-tad.com
Teléfono:	

2. DESCRIPCIÓN DE LA ASIGNATURA.

2.1 Descripción de la materia

Esta materia establece los conocimientos y técnicas necesarios para la correcta especificación, diseño e implementación de proyectos software atendiendo a las buenas prácticas y metodologías ingenieriles.

2.2 Descripción de la asignatura

Modelar y diseñar soluciones atendiendo a los compromisos de eficiencia, modularidad, calidad y mantenibilidad. Comprender y diseñar arquitecturas de software basadas en la orientación a objetos, mediante técnicas y patrones de diseño. Es una asignatura fundamental de cara a comprender cómo crear mejor software

3. COMPETENCIAS

Competencias Básicas y Generales
CB1: Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio que parte de la base de la educación secundaria general, y se suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados, incluye también algunos aspectos que implican conocimientos procedentes de la vanguardia de su campo de estudio. CB2: Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio. CB3: Que los estudiantes tengan la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética. CB4: Que los estudiantes puedan transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado. CB5: Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía. CG1 - Capacidad para entender, planificar y resolver problemas a través del desarrollo de soluciones informáticas. CG2 - Desarrollo de soluciones informáticas respetuosas con el medio ambiente, los deberes sociales y los recursos naturales, además de cumplir con la legislación y la ética CG3 - Conocimiento de los fundamentos científicos aplicables a la resolución de problemas informáticos CG6 - Integración, como ingeniero del software, en entornos de trabajo multidisciplinares demostrando capacidad de trabajo en equipo, versatilidad, flexibilidad, creatividad y respeto por el trabajo de los compañeros de otras áreas. CG9 - Capacidad para aprender, modificar y producir nuevas tecnologías informáticas CG10 - Uso de técnicas creativas para la realización de proyectos informáticos
Competencias Específicas
CE10 - Capacidad para manejar un gestor de versiones de código y generar la documentación de una aplicación de forma automática. CE18 - Capacidad para diseñar la arquitectura de una aplicación informática orientada a objetos empleando los patrones de diseño más adecuados e integrándolos en la arquitectura completa. CE19 - Capacidad para concebir, diseñar a través de lenguajes gráficos e implementar una aplicación informática empleando distintas metodologías de desarrollo, desde la concepción del producto hasta su desarrollo final pasando por la definición de sus fases e iteraciones CE20 - Capacidad para testar el funcionamiento y funcionalidad de una aplicación informática, elaborando planes de pruebas y empleando técnicas de diseño y programación orientado a las pruebas CE22 - Conocimiento de las técnicas e implicaciones del mantenimiento de aplicaciones informáticas incluyendo aquellas que utilizan principios de ingeniería inversa para entender y modificar un software cuya estructura se desconoce

4. CONTENIDOS

Tema 1_Lenguajes de descripción de software
Tema 2_Patrones de diseño • Patrón STRATEGY • Patrón OBSERVER • Patrón DECORATOR • Patrón STATE • Patrón FACTORY METHOD • Patrón ABSTRACT FACTORY • Patrón SINGLETON • Patrón ADAPTER • Patrón FACADE • Patrón TEMPLATE METHOD
Tema 3_Técnicas de refactorización

5. ACTIVIDADES FORMATIVAS Y MODALIDADES DE ENSEÑANZAS

5.1 Modalidades de enseñanza

La asignatura se desarrollará a través de los siguientes métodos y técnicas generales, que se aplicarán diferencialmente según las características propias de la asignatura:

- **Método expositivo/Lección magistral:** el profesor desarrollará, mediante clases magistrales y dinámicas los contenidos recogidos en el temario.
- **Estudio de casos:** análisis de casos reales relacionados con la asignatura.
- **Resolución de ejercicios y problemas:** los estudiantes desarrollarán las soluciones adecuadas aplicando procedimientos de transformación de la información disponible y la interpretación de los resultados.
- **Aprendizaje basado en problemas:** utilización de problemas como punto de partida para la adquisición de conocimientos nuevos.
- **Aprendizaje orientado a proyectos:** se pide a los alumnos que, en pequeños grupos, planifiquen, creen y evalúen un proyecto que responda a las necesidades planteadas en una determinada situación.
- **Aprendizaje cooperativo:** Los estudiantes trabajan en grupo para realizar las tareas de manera colectiva.

5.2 Actividades formativas

Actividad Formativa	Horas	Presencialidad
AF1 Clases teóricas / Expositivas	30	100%
AF2 Clases Prácticas	24	100%
AF3 Tutorías	6	50%
AF4 Estudio independiente y trabajo autónomo del alumno	57	0%
AF5 Elaboración de trabajos (en grupo o individuales)	29	0%
AF6: Actividades de Evaluación	4	100%

6. DESARROLLO TEMPORAL

UNIDADES DIDÁCTICAS / TEMAS	PERÍODO TEMPORAL
Tema 1_Lenguajes de descripción de software	Semana 1
Tema 2_Patrones de diseño: Introducción	Semana 2
Tema 2_Patrón STRATEGY	Semana 2, 3
Tema 2_Patrón OBSERVER	Semana 4, 5
Tema 2_Patrón DECORATOR	Semana 6
Tema 2_Patrón STATE	Semana 7
Tema 2_Patrón FACTORY METHOD	Semana 8
Tema 2_Patrón ABSTRACT FACTORY	Semana 9
Tema 2_Patrón SINGLETON	Semana 10
Tema 2_Patrón ADAPTER	Semana 11
Tema 2_Patrón FACADE	Semana 12
Tema 2_Patrón TEMPLATE METHOD	Semana 13
Tema 3_Técnicas de refactorización	Semanas 14 y 15

7. SISTEMA DE EVALUACIÓN

ACTIVIDAD DE EVALUACIÓN	VALORACIÓN MÍNIMA RESPECTO A LA CALIFICACIÓN FINAL (%)	VALORACIÓN MÁXIMA RESPECTO A LA CALIFICACIÓN FINAL (%)
SE1 Evaluación de la participación en clase, en prácticas o en proyectos de la asignatura	0%	30%
SE2 Evaluación de trabajos, proyectos, informes, memorias	30%	80%
SE3 Prueba Objetiva	10%	60%

8. CRITERIOS DE EVALUACIÓN

ACTIVIDAD DE EVALUACIÓN	VALORACIÓN RESPECTO A LA CALIFICACIÓN FINAL (%)
Evaluación de la participación en clase, y actividades de la asignatura	10%
Evaluación de trabajos, proyectos, informes, memorias	30%
Prueba Objetiva, Examen Final	60%

Consideraciones generales acerca de la evaluación:

. La evaluación se hará mediante un examen final en la convocatoria ordinaria donde será imprescindible sacar una nota mínima de 4 para poder hacer media con las notas del proyecto y de evaluación continua. El examen se realizará sin ordenador.

. La entrega de actividades y proyecto son opcionales, pero aquellos que no los entreguen sólo podrán sacar como máximo un 6 en la nota final de la asignatura, esto es, sacando un 10 en el examen.

. En cuanto a la convocatoria extraordinaria, el examen contará el 70% de la nota y el proyecto un 30% siendo de nuevo obligatorio obtener más de un 4 en el examen para hacer media. Se guardará la nota del proyecto si éste se ha realizado en la convocatoria ordinaria.

. No se admitirán trabajos fuera de forma y fecha sin causa justificada, cada entrega se entiende como un examen y tendrá derecho a revisión. Y si se aceptan será con una reducción considerable en la nota.

9. BIBLIOGRAFÍA / WEBGRAFÍA

Bibliografía Básica:

- Freeman, E.; Freeman, E.; Bates, B. y Sierra, K. (2004); HEAD FIRST DESIGN PATTERNS; Editorial. O'Reilly; ISBN: 0596007124
- Debrauwer,L. (2013); PATRONES DE DISEÑO EN JAVA: LOS 23 MODELOS DE DISEÑO: DESCRIPCION Y SOLUCION ILUSTRADAS EN UML 2 Y JAVA; Editorial: ENI; ISBN: 9782746086456
- Fowler, M. (1999); REFACTORING: IMPROVING THE DESIGN OF EXISTING CODE; Editorial: ADDISON-WESLEY; ISBN: 9780201485677

Bibliografía Recomendada:

- Gamma, E.; Johnson, R.; Helm, R. y Vlissides, J. (1994); DESIGN PATTERNS: ELEMENTS OF REUSABLE OBJECT-ORIENTED SOFTWARE; Addison-Wesley; ISBN: 0-201-63361-2
- Shalloway, Alan; Trott, James; Design Patterns Explained: A New Perspective on Object-Oriented. Addison-Wesley Professional (2001). ISBN 10: 0201715945 ISBN 13: 9780201715941
- Steven John Metsker; The design patterns Java workbook; Addison Wesley, 2002. ISBN**:** 0-201-74397-3
- James W. Cooper; Java™ Design Patterns: A Tutorial; Addison Wesley, 2000. ISBN: 0-201-48539-7

10. MATERIAL, SOFTWARE Y HERRAMIENTAS NECESARIAS

MATERIALES:

- Ordenador personal con Windows
- Eclipse