GUÍA DOCENTE

DISEÑO DE SOFTWARE

DOBLE GRADO EN INGENIERÍA DEL SOFTWARE

MODALIDAD: PRESENCIAL

CURSO ACADÉMICO: 2025-2026

Denominación de la asignatura:	Diseño de Software
Titulación:	Doble Grado en Ingeniería del Software
Facultad o Centro:	Centro Universitario de Tecnología y Arte Digital
Materia:	Ingeniería del Software
Curso:	2
Cuatrimestre:	2
Carácter:	OB
Créditos ECTS:	6
Modalidad/es de enseñanza:	Presencial
Idioma:	Castellano
Profesor/a - email	Miguel Angel Mesas Uzal / miguel.mesas@u-tad.com
Página Web:	http://www.u-tad.com/

DESCRIPCIÓN DE LA ASIGNATURA

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.

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

COMPETENCIAS Y RESULTADOS DE APRENDIZAJE DE LA MATERIA

Competencias (genéricas, específicas y transversales)

COMPETIENCIAS BÁSICAS Y GENERALES

CB1: Que los estudiantes hayan demostrado poseer y comprender conocimientosen un área de estudio que parte de la base de la educación secundaria general, yse suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados,incluye también algunos aspectos que implican conocimientos procedentes de lavanguardia de su campo de estudio.

CB2: Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocaciónde una forma profesional y posean las competencias que suelen demostrarse pormedio de la elaboración y defensa de argumentos y la resolución de problemasdentro de su área de estudio.

CB3: Que los estudiantes tengan la capacidad de reunir e interpretar datosrelevantes (normalmente dentro de su área de estudio) para emitir juicios queincluyan una reflexión sobre temas relevantes de índole social, científica o ética.

CB4: Que los estudiantes puedan transmitir información, ideas, problemas ysoluciones a un público tanto especializado como no especializado.

CB5: Que los estudiantes hayan desarrollado aquellas habilidades de aprendizajenecesarias para emprender estudios posteriores con un alto grado de autonomía.

CG1 - Capacidad para entender, planificar y resolver problemas a través deldesarrollo de soluciones informáticas.

CG2 - Desarrollo de soluciones informáticas respetuosas con el medio ambiente, losdeberes sociales y los recursos naturales, además de cumplir con la legislación y laética

CG3 - Conocimiento de los fundamentos científicos aplicables a la resolución deproblemas informáticos

CG6 - Integración, como ingeniero del software, en entornos de trabajomultidisciplinares 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 tecnologíasinformáticas

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 ladocumentación de una aplicación de forma automática.

CE18 - Capacidad para diseñar la arquitectura de una aplicación informáticaorientada a objetos empleando los patrones de diseño más adecuados eintegrándolos en la arquitectura completa.

CE19 - Capacidad para concebir, diseñar a través de lenguajes gráficos eimplementar una aplicación informática empleando distintas metodologías dedesarrollo, desde la concepción del producto hasta su desarrollo final pasando porla definición de sus fases e iteraciones

CE20 - Capacidad para testar el funcionamiento y funcionalidad de una aplicacióninformática, elaborando planes de pruebas y empleando técnicas de diseño yprogramación orientado a las pruebas

CE22 - Conocimiento de las técnicas e implicaciones del mantenimiento deaplicaciones informáticas incluyendo aquellas que utilizan principios de ingenieríainversa para entender y modificar un software cuya estructura se descono

Resultados de aprendizaje

Al acabar la titulación, el graduado o graduada será capaz de:

- Conocer los lenguajes de especificación formal

- Ser capaz de identificar y usar patrones de diseño en la resolución de problema

- Manejar las técnicas de refactorización

- Entender el ciclo de vida del software

- Entender y aplicar las metodologías waterfall en el desarrollo

- Entender y aplicar Scrum en el desarrollo

CONTENIDO

Lenguajes de descripción de software

Patrones de diseño

Técnicas de refactorización

TEMARIO

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

ACTIVIDADES FORMATIVAS Y METODOLOGÍAS DOCENTES

Actividades formativas

Actividad Formativa	Horas totales	Horas presenciales
Clases teóricas / Expositivas	25	25
Clases Prácticas	29	29
Tutorías	4	2
Estudio independiente y trabajo autónomo del alumno	50	0
Elaboración de trabajos (en grupo o individuales)	32	0
Actividades de Evaluación	10	10
Preparación y defensa del TFG	<<7- Preparación y defensa del TFG>>	<<Horas presenciales 7- Preparación y defensa del TFG>>

Metodologías docentes

Método expositivo o lección magistral

Aprendizaje de casos

Aprendizaje basado en la resolución de problemas

Aprendizaje cooperativo o colaborativo

Aprendizaje por indagación

Metodología Flipped classroom o aula invertida

Gamificación

Just in time Teaching (JITT) o aula a tiempo

Método expositivo o lección magistral

Método del caso

Aprendizaje basado en la resolución de problemas

Aprendizaje cooperativo o colaborativo

Aprendizaje por indagación

Metodología flipped classroom o aula invertida

Gamificación

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

SISTEMA DE EVALUACIÓN

ACTIVIDAD DE EVALUACIÓN	VALORACIÓN MÍNIMA RESPECTO A LA CALIFICACIÓN FINAL (%)	VALORACIÓN MÁXIMA RESPECTO A LA CALIFICACIÓN FINAL (%)

Evaluación de la participación en clase, en prácticas o en proyectos de la asignatura	0	30
Evaluación de trabajos, proyectos, informes, memorias	30	80
Prueba Objetiva	10	60
Evaluación del TFG	<<4-(MIN)Evaluación del TFG>>	0

CRITERIOS ESPECÍFICOS DE EVALUACIÓN

ACTIVIDAD DE EVALUACIÓN	CONVOCATORIA ORDINARIA	CONVOCATORIA EXTRAORDINARIA
Evaluación de la participación en clase, en prácticas o en proyectos de la asignatura	10	0
Evaluación de trabajos, proyectos, informes, memorias	30	30
Prueba Objetiva	60	70
Evaluación del TFG	<<4-(MIN)Evaluación del TFG>>	0

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.

. 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.

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

MATERIALES, SOFTWARE Y HERRAMIENTAS NECESARIAS

Tipología del aula
Aula teórica

Equipo de proyección y pizarra

Materiales:
Ordenador personal con Windows

Software:
Eclipse