Guía docente de la asignatura

Desarrollo de Software (Especialidad Ingeniería del Software) (296113F)

Fecha de aprobación: **26/06/2023**

Grado	Grado en Ingeniería Informática	**Rama**	Ingeniería y Arquitectura
Módulo	Formación de Especialidad 2: Ingeniería del Software	**Materia**	Desarrollo y Gestión de Proyectos

Curso	3º	**Semestre**	2º	**Créditos**	6	**Tipo**	Obligatoria

PRERREQUISITOS Y/O RECOMENDACIONES

Se recomienda la superación de los contenidos y adquisición de competencias de las materias de formación básica.

BREVE DESCRIPCIÓN DE CONTENIDOS (Según memoria de verificación del Grado)

- Patrones de diseño.
- Composición y clasificación de los patrones arquitectónicos.
- Arquitecturas orientadas a componentes y servicios.
- Técnicas de desarrollo dirigidas por modelos.
- Modelado de negocio.
- Técnicas de verificación y validación de software.
- Pruebas.
- Mantenimiento y evolución del software.

COMPETENCIAS ASOCIADAS A MATERIA/ASIGNATURA

COMPETENCIAS GENERALES

- CG01 - Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan por objeto, de acuerdo con los conocimientos adquiridos, la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas.
- CG05 - Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad.

COMPETENCIAS TRANSVERSALES

- CT03 - Capacidad para el uso y aplicación de las TIC en el ámbito académico y profesional.

RESULTADOS DE APRENDIZAJE (Objetivos)

- Conocer los patrones que se pueden aplicar a un diseño y a su estructura.
- Clasificación de los patrones y su importancia como herramienta para mejorar la calidad de un diseño.
- Adquirir destreza en la identificación de los patrones aplicables a un determinado problema.
- Comprender correctamente el concepto de componente-software y saber aplicarlo en le desarrollo.
- Conocer las técnicas de desarrollo dirigidas por modelos, sus ventajas e inconvenientes.
- Conocer y saber aplicar diferentes niveles de prueba a un software.
- Introducirse en el modelado de procesos de negocio, así como en sus técnicas fundamentales y herramientas.

PROGRAMA DE CONTENIDOS TEÓRICOS Y PRÁCTICOS

TEÓRICO

Tema 1. Desarrollo utilizando patrones software.

1. Análisis y diseño basado en patrones.
2. Estudio del catálogo GoF de patrones de diseño.
3. Patrones arquitectónicos.

Tema 2. Arquitectura Software.

1. Principios de Arquitectura Software.
2. Propiedades no funcionales de Arquitectura Software.
3. Descripción arquitectónica del software.

Tema 3. Pruebas software

1. Objetivos y principios fundamentales.
2. Validación y verificación.
3. Relación con el ciclo de vida del software.
4. Técnicas, herramientas y marcos de trabajo para el desarrollo de pruebas del software.

Tema 4. Desarrollo dirigido por modelos.

1. Fundamentos del desarrollo dirigido por modelos (MDD).
2. Arquitecturas dirigidas por modelos (MDA) y factorías de software.
3. Modelado de negocio.
4. Arquitecturas de empresa/Sistemas de Información Empresarial.

Tema 5. Mantenimiento y evolución del software.

1. Principios y tipos de mantenimiento aplicables al software.

2. El proceso de mantenimiento del software.
3. Actividades relacionadas con la correcta evolución del software.

Talleres

- Taller 1: Introducción a Git y GitHub.
- Taller 2: Introducción al desarrollo de apps con Flutter.
- Taller 3: Introducción al desarrollo de pruebas unitarias en Flutter.
- Taller 4: Introducción a RoR.
- Taller 5: Introducción a los servicios Web RESTful con RoR y uso con Flutter.

PRÁCTICO

- Práctica 1. Aplicar algunos patrones de diseño del catálogo GoF y arquitectónicos al diseño de varios programas que resuelven un conjunto de problemas propuestos. Analizar la adecuación del diseño e implementación obtenidos con respecto a los requisitos especificados para cada problema.
- Práctica 2. Implementar en Flutter una aplicación dentro de la temática del grupo pequeño de teoría, eligiendo el estilo arquitectónico más adecuado.
- Práctica 3. Desarrollo de pruebas unitarias y de integración para evaluar la calidad del software desarrollado en la práctica 2. Redacción de un informe de validación del software referido.
- Práctica 4. Realizar una propuesta de arquitectura software y aplicarla al desarrollo completo de una aplicación multiplataforma, incluyendo al menos una parte para móvil en Flutter, que parta de la práctica 2 y otra parte Web con Ruby on Rails y pruebas de unidad, de integración y de sistema.

BIBLIOGRAFÍA

BIBLIOGRAFÍA FUNDAMENTAL

- Gamma, E., Helm, E.,Johnson, R., Vlissides, J. Design Patterns. Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995. Traducción al español de César Fernández Acebal y Juan Manuel Cueva Lovelle. Pearson Educación, 2009.
- Nick Rozanski. Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives, Second Edition. Addison-Wesley Professional, 2011.
- Mary Shaw and David Garlan. Software Architecture. Prentice Hall, New Jersey, 1996.
- Daniel Galin. Software Quality. Wiley-IEEE Computer Society Press, 2018.
- Priyadarshi Tripathy and Kshirasagar Naik. Software Evolution and Maintenance. John Wiley, EE.UU., 2014.
- Alberto Rodrigues da Silva. Model-Driven Engineering: A survey supported by a unified conceptual model. Computer Languages, Systems & Structures, 20, 06 2015. doi: 10.1016/j.cl.2015.06.001.

BIBLIOGRAFÍA COMPLEMENTARIA

- Brad Appleton. Patterns and software: Essential concepts and terminology, 2000.
- Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. Pattern-Oriented Software Architecture - Volume 1: A System of Patterns. Wiley Publishing, 1996. ISBN 0471958697.
- Anand Balachandran Pillai. Software Architecture with Python. Packt Publishing, 2017.

- Len Bass, Paul Clements, and Rick Kazman. Software Architecture in Practice (2nd Edition). Addison-Wesley Professional, 2003.
- Judith Stafford, Robert Nord, Paulo Merson, Reed Little, James Ivers, David Garlan, Len Bass, Feliz Bachmann, and Paul Clements. Documenting Software Architectures: Views and Beyond, Second Edition. Addison-Wesley Professional, EE.UU., 2010.
- Heinz Züllighoven. Object-Oriented Construction Handbook. Science Direct, EE.UU., 2005.
- Michael Havey. Essential Business Process Modeling. O'Reilly Media, Inc., USA, 2005.
- Mika Koskela and Jyrki Haajanen. Business process modeling and execution. research notes 2407. Technical report, 2007.
- Stephen J. Mellor and Marc J. Balcer. Executable UML: A Foundation for Model-Driven Architecture. Addison-Wesley, USA, 2002.
- Stephen J. Mellor, K. Scott, Uhl A., and Weise D. MDA distilled: Principles of Model-Driven Architecture. Addison-Wesley Longman Publishing Co., Inc., USA, 2004.
- Alberto Rodrigues da Silva. Model-Driven Engineering: A survey supported by a unified conceptual model. Computer Languages, Systems & Structures, 20, 06 2015. doi: 10.1016/j.cl.2015.06.001.
- Jos Warmer, Anneke Kleppe, and Wim Bast. MDA Explained: The Model Driven Architecture: Practice and Promise. Addison-Wesley Professional, EE.UU., 2003.
- L. A. Belady and M. M. Lehman. A model of large program development. IBM Systems Journal, 1(15):225–252, 1976.
- Priyadarshi Tripathy and Kshirasagar Naik. Software Evolution and Maintenance. Johh Wiley, EE.UU., 2014.

ENLACES RECOMENDADOS

- O'Reilly
- PRADO
- Desarrollo de software

METODOLOGÍA DOCENTE

- MD01 - Lección Magistral (Clases Teóricas-Expositivas)
- MD02 - Actividades Prácticas (Resolución de Problemas, Resolución de Casos Prácticos, Desarrollo de Proyectos, Prácticas en Laboratorio, Taller de Programación, Aula de Informática, Prácticas de Campo).
- MD03 - Seminarios (Debates, Demos, Exposición de Trabajos Tutelados, Conferencias, Visitas Guiadas, Monografías).
- MD04 - Actividades no presenciales Individuales.
- MD05 - Actividades no presenciales Grupales.
- MD06 - Tutorías Académicas.

EVALUACIÓN (instrumentos de evaluación, criterios de evaluación y porcentaje sobre la calificación final)

EVALUACIÓN ORDINARIA

Todo lo relativo a la evaluación se regirá por la Normativa de evaluación y calificación de los

estudiantes vigente en la Universidad de Granada.
Preferentemente, la evaluación se ajustará al sistema de evaluación continua del aprendizaje del estudiante siguiendo el artículo 7 de la anterior Normativa.
En esta modalidad se utilizarán las siguientes técnicas de evaluación:

- Parte teórica básica: Para la parte teórica básica se realizará un trabajo o examen diferido (varios días para hacerlo) y un examen presencial.
- Otras actividades formativas: Se evaluarán asimismo actitudes a alcanzar como desarrolladores software, mediante otras actividades formativas: tareas individuales, tareas en grupo pequeño y los cuatro talleres. En los talleres se valorará la resolución de los problemas y ejercicios que hayan sido resueltos y entregados en plazo por los alumnos.
- Las prácticas de laboratorio se evaluarán mediante entrevistas al grupo pequeño. Se valorarán las entregas en fechas establecidas de los informes/memorias realizados para cada práctica.
- Para los estudiantes que se acojan a la evaluación única final, las pruebas y la evaluación estarán regidas por los criterios que se exponen más adelante en este documento (ver apartado correspondiente).
- La calificación global corresponderá a la puntuación obtenida de la ponderación de los diferentes aspectos y actividades que integran el sistema de evaluación. La adaptación del sistema de evaluación general propuesto a las características de esta asignatura se ajustará a lo indicado a continuación:

Porcentajes de evaluación

Actividades formativas	**Ponderación**
Parte teórica básica	30%
Parte práctica	50%
Otras actividades formativas (trabajos grupo pequeño, trabajos individuales y talleres)	20%

Para aprobar la asignatura es necesario obtener una calificación numérica superior o igual a 5 puntos (sobre 10). No obstante, además del requisito anterior, se establecen adicionalmente los siguientes requisitos:

- Parte teórica básica.- Se debe obtener una calificación mayor o igual a 4,5 puntos (sobre 10) en cada uno de los dos exámenes.
- Parte práctica.- Será necesario una calificación mayor o igual a 4,5 en cada una de las cuatro prácticas.
- Otras actividades formativas.- Se deberá obtener una puntuación de al menos 4,5 puntos para poder hacer media.

El sistema de calificaciones se expresará mediante calificación numérica de acuerdo con lo establecido en el artículo 5 del R.D. 1125/2003 de 5 de septiembre, por el que se establece el sistema europeo de créditos y el sistema de calificaciones en las titulaciones universitarias de carácter oficial y validez en el territorio nacional.

EVALUACIÓN EXTRAORDINARIA

La evaluación de los estudiantes en la convocatoria extraordinaria se regirá por los mismos criterios y constará de las mismas pruebas que las indicadas en este documento para evaluación única final.

EVALUACIÓN ÚNICA FINAL

La evaluación única final consistirá en dos pruebas de evaluación, una para la parte teórica y otra para la parte práctica, con las características que se indican a continuación:

- Evaluación de teoría: los estudiantes realizarán una prueba escrita que constará de preguntas de teoría, problemas y ejercicios sobre la teoría impartida y un trabajo extenso

o examen diferido (varios días para realizarlo) de resolución de supuestos más extensos.
- Evaluación de las prácticas: los estudiantes realizarán de manera individual la parte obligatoria de cada una de las prácticas de laboratorio de este curso y una prueba en el laboratorio que consistirá en la realización de modificaciones de dichas prácticas a partir de cambios en los requisitos de las mismas.

La ponderación de cada parte en la nota final será del 50% (parte teórica) y 50% (parte práctica).
Para aprobar la asignatura se han de cumplir los siguientes requisitos:
- La nota de la prueba de teoría ha de ser igual o superior al 4,5.
- La nota de la prueba de prácticas ha de ser igual o superior a 4,5.
- La suma ponderada de ambas notas ha de ser igual o superior a 5.

INFORMACIÓN ADICIONAL

Competencias básicas a destacar en esta asignatura
- 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.

Competencias Generales del título a destacar en esta asignatura
- E5. Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad.
- E8. Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.
- E9. Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática.
- E11. Capacidad para analizar y valorar el impacto social y medioambiental de las soluciones técnicas, comprendiendo la responsabilidad ética y profesional de la actividad del Graduado en Informática.

Competencias transversales a destacar en esta asignatura:
- T1. Capacidad de organización y planificación así como capacidad de gestión de la Información.
- T2. Capacidad para tomar decisiones basadas en criterios objetivos (datos experimentales, científicos o de simulación. disponibles) así como capacidad de argumentar y justificar lógicamente dichas decisiones, sabiendo aceptar otros puntos de vista.
- T4. Capacidad de comunicación en lengua extranjera, particularmente en inglés.
- T5. Capacidad de trabajo en equipo, usando competencias demostrables mediante la elaboración y defensa de argumentos.
- T6. Motivación por la calidad y la mejora continua, actuando con rigor, responsabilidad y ética profesional.
- T7. Respeto a los derechos fundamentales y de igualdad entre hombres y mujeres.
- T8. Capacidad para proyectar los conocimientos, habilidades y destrezas adquiridos para promover una sociedad basada en los valores de la libertad, la justicia, la igualdad y el pluralismo.

Competencias generales
- CG01 - Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar

proyectos en el ámbito de la ingeniería en informática que tengan por objeto, de acuerdo con los conocimientos adquiridos, la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas.
- CG05 - Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad.

Metodología docente

1. Clases de teoría (grupo grande de teoría)
    - Descripción: Resumen de los contenidos que deben adquirirse, exposición de aquéllos más importantes o más difíciles de comprender, exposición de la resolución de problemas y casos prácticos realizada en grupos pequeños.
    - Propósito: Adquisición de un conocimiento profundo de la importancia del desarrollo del software, desarrollando la capacidad de reflexión, descubrimiento de las relaciones entre diversos conceptos, mentalidad crítica y presentación pública.
    - Contenido en ECTS: 13 horas presenciales (0,52 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T5, T6.
    - Metodología empleada: explicación del profesor ("píldoras" de 10-15 minutos), exposición de los grupos pequeños.
2. Actividades presenciales grupales (grupo pequeño de teoría)
    - Descripción: Actividades propuestas por el profesor a través de las cuales y de forma grupal se profundiza en aspectos concretos de la materia posibilitando a los estudiantes avanzar en la adquisición de determinados conocimientos y procedimientos de la materia. Se incluye la puesta en común de cuestiones de un calado más profundo, relacionados con las implicaciones éticas y la necesidad de formación humanística-teológica como complemento universitario a la formación tecnológica.
    - Propósito: Favorecer en los estudiantes las capacidades (1) de trabajo en equipo, (2) de identificación y análisis de diferentes puntos de vista sobre una temática sabiendo reconocer las distintas responsabilidades de cada miembro de un equipo de desarrollo software para resolver lo conflictos en casos en los que no exista consenso de opiniones, (3) de generalización o transferencia de conocimiento, (4) de valoración crítica del mismo, (5) y de autocrítica.
    - Contenido en ECTS: 13 horas presenciales (0,52 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T5, T6, T8.
    - Metodologías docentes: Trabajo en grupo pequeño de teoría como equipo de desarrollo software. Se trabajará con ordenadores personales en el aula de teoría o en cualquier otro lugar en el que el grupo pueda mantener conexión remota -videoconferencia- con el profesor.
3. Talleres (grupo grande de teoría)
    - Descripción: Aprendizaje práctico introductorio de herramientas concretas usadas en el Desarrollo Software.
    - Propósito: Facilitar el inicio de aprendizaje de herramientas concretas a usar en la asignatura, y sobre las que cada estudiante deberá profundizar en su conocimiento y destreza de uso a lo largo del curso.
    - Contenido en ECTS: 4 horas presenciales (0,16 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T5, T6.
    - Metodologías docentes: Taller de Programación (aula de teoría o presencialidad remota -videoconferencia- con ordenadores personales).

4. Clases tutorizadas y presentación de prácticas de laboratorio (grupo pequeño y grupo grande de prácticas)
    - Descripción: Inicio de cada sesión semanal en los equipos software de prácticas (grupo pequeño de prácticas) bajo la supervisión del profesor y presentación de prácticas ya realizadas.
    - Propósito: Planificar la tarea a realizar en esa sesión repartiendo el trabajo entre los componentes del equipo, comienzo de la misma y resolución de todas las dudas que surjan para poder completarlas de forma autónoma. Presentación de la práctica realizadas al profesor (prácticas 1, 2 y 3) o a todo el grupo (práctica 4).
    - Contenido en ECTS: 25 horas presenciales (1 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T5, T6.
    - Metodologías empleadas: Trabajo en grupo pequeño, resolución de dudas por parte de los compañeros del equipo y, si aún permanecen, por parte del profesor. Presentación mediante demostraciones, diapositivas, documentos de texto, tutoriales, etc.
5. Clases autónomas de prácticas de laboratorio (grupo pequeño de prácticas)
    - Descripción: Realización de las prácticas en los equipos software de prácticas (grupo pequeño de prácticas).
    - Propósito: Realización completa de la práctica y entrega de la misma.
    - Contenido en ECTS: 50 horas presenciales (2 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T5, T6.
    - Metodologías empleadas: Distintas herramientas de desarrollo software, incluyendo software de control de versiones para trabajo en equipo.
6. Actividades no-presenciales individuales (estudio y trabajo autónomo)
    - Descripción: 1) Actividades (guiadas y no guiadas) propuestas por el profesor a través de las cuales y de forma individual se profundiza en aspectos concretos de la materia posibilitando al estudiante avanzar en la adquisición de determinados conocimientos y procedimientos de la materia; 2) Estudio individualizado de los contenidos de la materia; 3) Actividades auto-evaluativas (informes, exámenes, ...); 4) Cuestionarios de reflexión profunda (implicaciones éticas, necesidad de formación humanística y teológica como fundamento para la formación tecnológica)
    - Propósito: Favorecer en el estudiante la capacidad para autorregular su aprendizaje, planificándolo, diseñándolo, evaluándolo y adecuándolo a sus especiales condiciones e intereses.
    - Contenido en ECTS: 39 horas no presenciales (1.56 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T5.
7. Tutorías académicas (individual o grupo pequeño de teoría o de prácticas)
    - Descripción: manera de organizar los procesos de enseñanza y aprendizaje que se basa en la interacción directa entre el estudiante y el profesor.
    - Propósito: 1) Orientan el trabajo autónomo y grupal del alumnado; 2) profundizar en distintos aspectos de la materia y 3) orientar la formación académica-integral del estudiante.
    - Contenido en ECTS: 1 hora presencial (incluida videoconferencia), grupales e individuales (0.04 ECTS).
    - Competencias: IS1, IS3, IS4, IS6, E1, E2, E3, E4, E5, E6, E7, E9, E10, E12, T1, T2, T4, T6.

Definición de grupo grande y pequeño:

- Los grupos grandes son los grupos de teoría o de prácticas
- Los grupos pequeños de teoría y prácticas de laboratorio son grupos de 3 a 4 estudiantes.