Guía docente de la asignatura

Metodologías y Herramientas para el Desarrollo Evolutivo de Software (M52/56/2/24)

Fecha de aprobación por la Comisión Académica: **15/07/2022**

Máster	Máster Universitario en Desarrollo del Software
MÓDULO	Módulo 1: Ingeniería del Software Avanzada
RAMA	Ingeniería y Arquitectura
CENTRO RESPONSABLE DEL TÍTULO	Escuela Internacional de Posgrado

Semestre	Anual	**Créditos**	4	**Tipo**	Optativa	**Tipo de enseñanza**	Semipresencial

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

El desarrollo de software es un proceso iterativo en el que el producto cambia con extrema facilidad. Tradicionalmente los enfoques de desarrollo para el mantenimiento han perseguido incrementar la mantenibilidad del software. Numerosas tendencias apuntan actualmente a considerar el mantenimiento como una parte concreta de un fenómeno de mayor envergadura teórica y práctica: la evolución del software.

Nuestro interés en este curso es aunar el desarrollo orientado a objetos con una visión del desarrollo como proceso evolutivo, de forma que todo el proceso se dirija hacia la inevitabilidad, versatilidad e integración del cambio. Así, el contenido teórico del curso se divide en seis puntos principales:

- Conceptos previos,
- La evolución en el desarrollo orientado a objetos,
- Evolución en los procesos de desarrollo,
- Mecanismos de evolución en programación y diseño,
- Arquitecturas del software y evolución, y
- Mecanismos de evolución en especificación.

Software development is an iterative process in which the product changes easily. Traditionally,

development approaches to maintenance are focused on increasing software maintainability. Numerous trends now point to considering maintenance as a concrete part of a phenomenon of greater theoretical and practical scope: the evolution of software.

Our interest in this course is to combine object-oriented development with a vision of software development as an evolutionary process, so that the entire process is directed towards the inevitability, versatility and integration of change. Thus, the theoretical content of the course is divided into six main points:

- Previous concepts,
- Evolution in object-oriented development,
- Evolution in the development processes,
- Mechanisms of evolution in programming and design,
- Software architectures and evolution, and

Mechanisms of evolution in specification.

COMPETENCIAS

COMPETENCIAS BÁSICAS

- CB6 - Poseer y comprender conocimientos que aporten una base u oportunidad de ser originales en desarrollo y/o aplicación de ideas, a menudo en un contexto de investigación.
- CB7 - Que los estudiantes sepan aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios (o multidisciplinares) relacionados con su área de estudio.
- CB8 - Que los estudiantes sean capaces de integrar conocimientos y enfrentarse a la complejidad de formular juicios a partir de una información que, siendo incompleta o limitada, incluya reflexiones sobre las responsabilidades sociales y éticas vinculadas a la aplicación de sus conocimientos y juicios.
- CB9 - Que los estudiantes sepan comunicar sus conclusiones y los conocimientos y razones últimas que las sustentan a públicos especializados y no especializados de un modo claro y sin ambigüedades.
- CB10 - Que los estudiantes posean las habilidades de aprendizaje que les permitan continuar estudiando de un modo que habrá de ser en gran medida autodirigido o autónomo.

COMPETENCIAS GENERALES

- CG01 - Habilidades cognitivas: conocer los principales problemas o retos tecnológicos planteados en el ámbito de las líneas de investigación del programa de posgrado, conocer los principios de las técnicas o metodologías de solución para dichos problemas propuestas por la comunidad científica, conocer las debilidades y fortalezas de dichas soluciones, así como conocer las aplicaciones que este conocimiento tiene en la sociedad actual.
- CG02 - Destreza para iniciar un trabajo de investigación científica o desarrollo tecnológico original e innovador, en el marco de los problemas descritos en el punto anterior.
- CG03 - Ser capaz de emplear el conocimiento científico existente en la resolución de problemas o mejora de procesos a nivel individual o en el contexto de empresas u

organismos públicos.
- CG04 - Capacidades sistémicas para obtener la capacidad de asimilación y adaptación a la evolución futura del estado del arte en el ámbito de las disciplinas científicas del Máster.
- CG05 - Destrezas tecnológicas: capacidad de usar, evaluar, crear, modificar o extender la herramientas informáticas útiles en la resolución de problemas relacionados con las líneas de investigación
- CG06 - Capacidades metodológicas: conocer las principales fuentes bibliográficas que describen los avances científicos en las líneas de investigación del programa de posgrado.
- CG07 - Destrezas lingüísticas: conocer y utilizar la terminología científica especializada, tanto en español como en inglés, relacionada con las líneas de investigación del departamento.
- CG08 - Competencias personales: capacidad de análisis y síntesis en la resolución efectiva de problemas, así como capacidad de toma de decisiones, organización y planificación. Capacidad de comunicación escrita y oral.
- CG09 - Competencias interpersonales: capacidad de trabajo en equipo, incluyendo la toma de decisiones en colectivos o grupos. Habilidades en las relaciones interpersonales. Habilidades para presentar trabajos y mantener debates en grupo.
- CG10 - Destrezas de redacción: ser capaz de expresar los resultados y el desarrollo de las investigaciones en textos o informes científico-técnicos, conocer los mecanismos de revisión entre pares propios de la ciencia para estos documentos, así como los mecanismos para su difusión en forma de artículos en revistas, libros, sitios web o en aportaciones a congresos.

COMPETENCIAS ESPECÍFICAS

- CE01 - Ser capaz de llevar a cabo un trabajo de investigación en campos científicos relacionados con el desarrollo del software, teniendo en cuenta los recursos disponibles y sus implicaciones éticas y sociales
- CE03 - Identificar y comprender los conceptos clave y las principales características de los sistemas software, en cuanto a sus requerimientos, diseño o programación, así como saber aplicar los principales modelos, métodos y técnicas de la Ingeniería del Software al desarrollo de estos sistemas.
- CE04 - Conocer y saber aplicar métodos, técnicas y herramientas avanzadas de modelado, análisis, diseño y simulación en sistemas colaborativos, ubicuos, móviles, distribuidos, de diálogo, empotrados, de tiempo real o de procesos de negocio.
- CE05 - Identificar y valorar propiedades software de usabilidad, accesibilidad, seguridad, confiabilidad, rendimiento, y ética informática, entre otras, y analizar cómo afectan a la calidad de un sistema software.
- CE06 - Saber aplicar las estrategias de modelado más adecuadas para el diseño de sistemas software, así como las técnicas para la generación sistemática de sistemas dirigido por modelos
- CE07 - Diseñar y desarrollar sistemas software desde una perspectiva centrada en el usuario.
- CE08 - Diseñar modelos de sistemas software que permitan aplicar mecanismos evolutivos de reflexión, parametrización, refactorización, reutilización y simulación de procesos, entre otros.
- CE09 - Conocer los paradigmas, fundamentos y técnicas específicas de interacción persona-ordenador para el diseño de sistemas software de interacción multimodales (voz, tangibles, gestos)
- CE10 - Comprender las metodologías y técnicas asociadas al desarrollo e implantación de aplicaciones web, en cuanto al sistema hipermedia construido, al gestor de contenido seleccionado, o la tecnología de desarrollo web utilizada en su implementación, así como

comprender las diferencias existentes en cuanto al diseño y desarrollo frente a otros tipos de aplicaciones.
- CE12 - Comprender y conocer técnicas de representación, interconexión, implementación, despliegue, y reutilización de servicios y componentes software y de negocio para su aplicación en sistemas colaborativos, distribuidos, ubicuos, empotrados y/o de tiempo real.

COMPETENCIAS TRANSVERSALES

- CT01 - Mostrar interés por la calidad y la excelencia en la realización de diferentes tareas.
- CT02 - Comprender y defender la importancia que la diversidad de culturas y costumbres tienen en la investigación o práctica profesional.
- CT03 - Tener un compromiso ético y social en la aplicación de los conocimientos adquiridos.
- CT04 - Ser capaz de trabajar en equipos interdisciplinarios para alcanzar objetivos comunes desde campos expertos diferenciados.
- CT05 - Incorporar los principios del Diseño Universal en el desempeño de su profesión.

RESULTADOS DE APRENDIZAJE (Objetivos)

El alumno sabrá/comprenderá:

- Identificar las necesidades específicas de evolución que surgen en cada etapa de desarrollo de un sistema software.
- Evaluar la utilidad de aplicar técnicas de refactorización durante la especificación del software.
- Aplicar mecanismos de evolución en programación, como por ejemplo: meta-clases y reflexión.
- Aplicar mecanismos de evolución en diseño, como por ejemplo: componentes y reutilización.
- Aplicar mecanismos de evolución en especificación, como por ejemplo: evolución de esquemas y objetos.
- Construir sistemas parametrizables.
- Construir sistemas basados en modelos.
- Construir sistemas evolutivos.

El alumno será capaz de:

- Realizar el esfuerzo de abstracción necesario para prever y diseñar la evolución del software.
- Realizar el esfuerzo de ingeniería necesario para hacer buenos modelos (y meta-modelos) del software.
- Aprender a jugar el rol que toca en cada momento de un proyecto software.
- Utilizar patrones de diseño y patrones arquitectónicos en el diseño de un sistema software.
- Elegir la mejor metodología para construir todo o parte de un sistema software concreto.

PROGRAMA DE CONTENIDOS TEÓRICOS Y PRÁCTICOS

TEÓRICO

1. Visión general de la evolución de software: Conceptos generales de evolución de sistemas. Evolución en el desarrollo Orientado de Objeto. Evolución y proceso de desarrollo de software

2. Mecanismos de evolución en programación y diseño: Estado del arte. Metaclases. Reflexión. Patrones de diseño. Evolución web.

3. Arquitecturas del software y evolución: Arquitecturas software. Arquitecturas dinámicas. Componentes y SOA (Service Oriented Architecture). Arquitectura evolutiva basada en agentes. Arquitecturas móviles.

4. Mecanismos de evolución en especificación: Nivel meta y desarrollo basados en modelos. Nivel meta y sistemas parametrizables.

---

1. Overview of software evolution: General concepts of systems evolution. Evolution in Object Oriented Development. Evolution and software development process.

2. Mechanisms of evolution in programming and design: State of the art. Metaclasses. Reflection. Design patterns. Web evolution.

3. Software architectures and evolution: Software architectures. Dynamic architectures. Components and SOA (Service Oriented Architecture). Evolutionary architecture based on agents. Mobile architectures.

4. Mechanisms of evolution in specification: Meta level and development based on models. Meta level and parametrizable systems.

PRÁCTICO

BIBLIOGRAFÍA

BIBLIOGRAFÍA FUNDAMENTAL

Se recomiendan las siguientes obras clásicas para adquirir los conocimientos previos necesarios:

Programación orientada a objetos:

- Budd, T. (1994) “Introducción a la programación orientada a objetos”. Addison-Wesley Iberoamericana.
- Brauer, J. (2015). Programming Smalltalk-Object-Orientation from the Beginning. Springer.
- Reges, S., & Stepp, M. (2014). Building Java Programs. Pearson.
- Medina-medina, N. (2015). Programación Orientada a Objetos con JAVA. La novela. RC.

Ingeniería del software:

- Pressman, R. “Ingeniería del Software: un enfoque práctico. McGraw-Hill” (cualquiera de las últimas ediciones)

UML (por este orden)

1. Rumbaugh, J. et al. "Modelado y diseño orientado a objetos". Prentice- Hall. 1996.
2. Larman, C. "UML y patrones: Introducción al análisis y diseño orientado a objetos". Prentice halll. 1999 ó 2003.
3. Booch, G. et al. (1999).. "El Lenguaje Unificado de Modelado: UML" . Addison Wesley. 1999.
4. Booch G. et al. "El proceso unificado de desarrollo de software". Addison Wesley. 2000.
5. Weilkiens, T. (2011). Systems engineering with SysML/UML: modeling, analysis, design. Elsevier.

BIBLIOGRAFÍA COMPLEMENTARIA

- Ampatzoglou, A., Chatzigeorgiou, A., Charalampidou, S., & Avgeriou, P. (2015). The effect of GoF design patterns on stability: a case study. IEEE Transactions on Software Engineering, 41(8), 781-802.
- Beck, K. The Inevitability of Evolution, IEEE Software, 27(4), 28-29, 2010
- Berrocal J.; Garcia-Alonso J.; Murillo J.M.; Canal C.; Rich contextual information for monitoring the elderly in an early stage of cognitive impairment, Pervasive and Mobile Computing, Volume 34, January 2017, Pages 106-125, ISSN 1574-1192, http://dx.doi.org/10.1016/j.pmcj.2016.05.001.
- Berrocal J.; Garcia-Alonso J.; Vicente-Chicote C.; Hernández J.; Mikkonen T.; Canal C.; Murillo J.M. Early analysis of resource consumption patterns in mobile applications, Pervasive and Mobile Computing, Volume 35, February 2017, Pages 32-50, ISSN 1574-1192, http://dx.doi.org/10.1016/j.pmcj.2016.06.011.
- Boehm B. The Changing Nature of Software Evolution. IEEE Software, 27(4), 26-28, 2010
- Buschmann, F. “Pattern-Oriented Software Architecture, Volume 1: A System of Patterns”. John Wiley and Sons, Inc. New York, NY.1996
- Canal, C., Fuentes, L., Pimentel, E., Troya, J.M.. “Coordinación de Componentes Distribuidos: un Enfoque Generativo Basado en Arquitectura del Software”. IV Jornadas de Ingeniería del Software y Bases de Datos (JISBD’99). Pp: 443-454. 1999.
- Capilla, R., Jansen, A., Tang, A., Avgeriou, P., & Babar, M. A. (2016). 10 years of software architecture knowledge management: Practice and future. Journal of Systems and Software, 116, 191-205.
- Cha, S., Taylor, R. N., & Kang, K. (Eds.). (2019). Handbook of software engineering. Springer International Publishing.
- Clark, T. (2018). Metaclasses and Reflection in Smalltalk. arXiv preprint arXiv:1804.07272.
- Feng Chen; Hongji Yang; Hong Zhou; Bing Qiao; Huifang Deng; , "Web-based system evolution in model driven architecture," Web Site Evolution, 2008. WSE 2008. 10th International Symposium on , vol., no., pp.69-72, 3-4 Oct. 2008
- Filippo, R., Liu., C. Special section on web systems evolution. International Journal Software Tools Technology Trans (STTT). , 11 (6) (2009), pp. 419–425
- Fowler, M. (2018). Refactoring: improving the design of existing code. Addison-Wesley Professional.
- Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. “Design Patterns: Elements of Reusable Object-Oriented Software”. First ed: Addison-Wesley Publishing Company. ISBN:0201633612. 1995
- García-Cabrera, Medina-Medina, N. "An axiomatic approach to maintaining the consistency of a hypermedia information system based on the SEM-HP model." Logic

Journal of IGPL 22.6, 1045-1074. 2014
- Garrido, A., Rossi, G., Medina, N. M., Grigera, J., & Firmenich, S. Improving accessibility of Web interfaces: refactoring to the rescue. Universal access in the information society, 13(4), 387-399. 2014
- Garrido, G. Rossi and D. Distante. "Refactoring for Usability in Web Applications". IEEE Software 28 (3), 60-67. May/June 2011.
- Godfrey, M.W., German, D.M. "The past, present, and future of software evolution," Frontiers of Software Maintenance, 2008. FoSM 2008. , vol., no., pp.129-138, Sept. 28 2008-Oct. 4 2008
- Grand, M. "Patterns in Java, a catalogue of reusable design patterns illustrated with UML". Volúmenes 1 y 2. Indianapolis, Indiana Wiley Publishing. ISBN: 0-471-22729-3. 2002
- Hassan, A., Queudet, A., & Oussalah, M. (2016, November). Evolution style: framework for dynamic evolution of real-time software architecture. In European Conference on Software Architecture (pp. 166-174). Springer, Cham.
- Herraiz, I., Rodriguez, D., Robles, G., & Gonzalez-Barahona, J. M. (2013). The evolution of the laws of software evolution: A discussion based on a systematic literature review. ACM Computing Surveys (CSUR), 46(2), 1-28.
- Kienle, H., Di Lucca, G., Tilley, S. Research Directions in Web Systems Evolution IV: Migrating to the Cloud. 12th IEEE International Symposium on Web Systems Evolution (WSE), 22, 2010
- Mäkitalo N.; Pääkkö J.; Raatikainen M.; Myllärniemi V.; Aaltonen T.; Leppänen T.; Männistö T.; Mikkonen T. Social devices: Collaborative co-located interactions in a mobile cloud, in: Proc. 11th Intl Conf. Mobile and Ubiquitous Multimedia, 2012.
- Marchetto, A., Ricca, F.: From objects to services: toward astepwise migration approach for java applications. International Journal on Software Tools For Technology Transfer (Special section on Web Systems Evolution) 11(6), 2009.
- Mehboob, Z.; Zowghi, D.; Lowe, D. "An Approach for Comparison of Architecture Level Change Impact Analysis Methods and Their Relevance in Web Systems Evolution," Software Engineering Conference, 2009. ASWEC '09. Australian , vol., no., pp.162-172,14-17 April 2009
- Mens, Tom; Guehéneuc, Yann-Gaël; Fernández-Ramil, Juan; D'Hondt, Maja; , "Guest Editors' Introduction: Software Evolution," Software, IEEE , vol.27, no.4, pp.22-25, July-Aug. 2010
- Mesbah A.: Ajaxifying classic web applications. In L:Proceedings of the 29th International Conference on Software Engineering (ICSE'07 Companion), Doctoral Symposium. pages 81–82. IEEE Computer Society, 2007.
- Meyer, B. "The Significance of Components. Beyond Objects" Software Development. Vol. 7(11). 1999
- Miranda J.; Mäkitalo N.; Garcia-Alonso J.; Berrocal J.; Mikkonen T.; Canal C.; Murillo J.M. From the internet of things to the internet of people, IEEE Internet Comput. 19 (2) (2015) 40–47
- Navarro, E., Cuesta, C., Dewayne E. P., Roda, C.: Using Model Transformation Techniques for the Superimposition of Architectural Styles. ECSA 2011: 379-387
- Paderewski Rodríguez, P. Un modelo arquitectónico evolutivo para sistemas software basados en agentes (Doctoral dissertation, Universidad de Granada). 2003
- Schmid, H. A. "Systematic Framework Design by Generalization" ACM, Special issue on Object Oriented Application frameworks. Vol. 40(10).1997
- Smaragdakis, Y., Balatsouras, G., Kastrinis, G., & Bravenboer, M. (2015, November). More sound static handling of Java reflection. In Asian Symposium on Programming Languages and Systems (pp. 485-503). Springer, Cham.
- Szyperski, C.; Gruntz, D.; Murer, S. "Component Software: Beyond Object-Oriented Programming". Addison-Wesley. 2002

ENLACES RECOMENDADOS

Como apoyo a la docencia se usará la Plataforma de Recursos de Apoyo a la Docencia PRADO de la Universidad de Granada: https://pradoposgrado2022.ugr.es

METODOLOGÍA DOCENTE

- MD01 Lección magistral/expositiva
- MD02 Sesiones de discusión y debate
- MD03 Resolución de problemas y estudio de casos prácticos
- MD05 Seminarios
- MD07 Análisis de fuentes y documentos
- MD08 Realización de trabajos en grupo
- MD09 Realización de trabajos individuales

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

EVALUACIÓN ORDINARIA

El artículo 18 de la Normativa de Evaluación y Calificación de los Estudiantes de la Universidad de Granada establece que la convocatoria ordinaria estará basada preferentemente en la evaluación continua del estudiante, excepto para quienes se les haya reconocido el derecho a la evaluación única final.

Se realizará una evaluación continua del trabajo del estudiante, valorando tanto los conocimientos adquiridos como las competencias alcanzadas.

Modalidad semipresencial:

Para la evaluación en modalidad semipresencial se tendrán en cuenta los siguientes sistemas de evaluación, indicándose entre paréntesis el rango del porcentaje con respecto a la calificación final del estudiante:

1. Asistencia a clase de los estudiantes: tendrá un peso del 25% de la calificación final.
2. Participación activa durante la impartición del curso: interés del estudiante, respuesta a las preguntas planteadas por el profesor durante la sesión, correcta realización de ejercicios planteados en el aula, etc.: tendrá un peso del 25% de la calificación final.
3. Participación activa en los debates virtuales utilizando la plataforma: tendrá un peso del 10% de la calificación final.
4. Entrega en tiempo y forma de las actividades propuestas para esta modalidad a través de la plataforma: tendrá un peso del 20% de la calificación final.
5. Realización de un trabajo final optativo donde se recojan uno o varios de los temas vistos durante el curso: tendrá un peso del 20% de la calificación final.

Se pedirá la entrega en tiempo y forma de las actividades propuestas a través de la plataforma PRADO.

Modalidad virtual:

1. Participación activa en los debates virtuales utilizando la plataforma: tendrá un peso del 20% de la calificación final.
2. Entrega en tiempo y forma de las actividades propuestas para esta modalidad a través de la plataforma: tendrá un peso del 50% de la calificación final.
3. Realización de un trabajo final optativo donde se recojan uno o varios de los temas vistos durante el curso: tendrá un peso del 30% de la calificación final.

Se pedirá la entrega en tiempo y forma de las actividades propuestas a través de la plataforma PRADO.

Aunque los criterios de evaluación en las dos modalidades son similares, las actividades a realizar para cada modalidad pueden diferir, al contar con herramientas distintas tanto para la adquisición de los conocimientos como para la evaluación.

A continuación se especifican las actividades formativas previstas así como su temporalización dependiendo de la modalidad de estudio:

		Modalidad Semipresencial		Modalidad virtual
Actividades formativas	Hora lectivas	Horas presenciales	Presencialidad	Horas virtuales
Clases teóricas	20	10	50%	20
Clases prácticas	4	2	50%	4
Trabajos Tutorizados	6	3	50%	6
Tutorías	6	3	50%	6
Evaluación	4	2	50%	4
Trabajo Autónomo	60	60		60
Total	100	80		100

Se llevarán a cabo sesiones orales para el control, evaluación y seguimiento de todos los alumnos.

Se realizará el seguimiento de las incidencias y dificultades que tengan los estudiantes en la modalidad presencial y virtual a través de la herramienta PRADO.

EVALUACIÓN EXTRAORDINARIA

El artículo 19 de la Normativa de Evaluación y Calificación de los Estudiantes de la Universidad de Granada establece que los estudiantes que no hayan superado la asignatura en la convocatoria ordinaria dispondrán de una convocatoria extraordinaria. A ella podrán concurrir todos los estudiantes, con independencia de haber seguido o no un proceso de evaluación continua. De esta forma, el estudiante que no haya realizado la evaluación continua tendrá la posibilidad de obtener el 100% de la calificación mediante la realización de una prueba y/o trabajo.

La evaluación en tal caso consistirá en la realización de una prueba y/o trabajo, y/o las actividades propuestas en la evaluación continua.

EVALUACIÓN ÚNICA FINAL

El artículo 8 de la Normativa de Evaluación y Calificación de los Estudiantes de la Universidad de Granada establece que podrán acogerse a la evaluación única final, el estudiante que no pueda cumplir con el método de evaluación continua por causas justificadas.

Para acogerse a la evaluación única final, el estudiante, en las dos primeras semanas de impartición de la asignatura o en las dos semanas siguientes a su matriculación si ésta se ha producido con posterioridad al inicio de las clases o por causa sobrevenidas. Lo solicitará, a través del procedimiento electrónico, a la Coordinación del Máster, quien dará traslado al profesorado correspondiente, alegando y acreditando las razones que le asisten para no poder seguir el sistema de evaluación continua.

La evaluación en tal caso consistirá en la realización de una prueba y/o trabajo, y/o las actividades propuestas en la evaluación continua.