Guía docente
340379 - AMEP-I4O23 - Ampliación a la Ingeniería del Programario

Última modificación: 28/01/2025

Unidad responsable: Escuela Politécnica Superior de Ingeniería de Vilanova i la Geltrú
Unidad que imparte: 747 - ESSI - Departamento de Ingenieria de Servicios y Sistemas de Información.

Titulación: GRADO EN INGENIERÍA INFORMÁTICA (Plan 2018). (Asignatura obligatoria).

Curso: 2024 **Créditos ECTS:** 6.0 **Idiomas:** Catalán

PROFESORADO

Profesorado responsable: Xavier Franch Gutiérrez

Otros: Xavier Franch Gutiérrez
Lidia López Cuesta

CAPACIDADES PREVIAS

- Conocimiento de los principios básicos de la Ingeniería del Software (ES)
- Capacidad de especificar, diseñar e implementar programas no triviales aplicando una metodología basada en planes (UML)
- Capacidad de participar en un proyecto de desarrollo de software en equipos pequeños

REQUISITOS

Introducción a la Ingeniería del Software (INEP)

COMPETENCIAS DE LA TITULACIÓN A LAS QUE CONTRIBUYE LA ASIGNATURA

Específicas:
1. CEFC12. Conocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos.
2. CEFC13. Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas de información, incluidos los basados en web.
3. CEFC16. Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.
4. CEFC6. Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.
5. CEIS1. Capacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y prácticas de la Ingeniería del Software.
6. CETI2. Capacidad para seleccionar, diseñar, desplegar, integrar, evaluar, construir, gestionar, explotar y mantener las tecnologías de hardware, software y redes, dentro de los parámetros de coste y calidad adecuados.

Transversales:
7. APRENDIZAJE AUTÓNOMO - Nivel 1: Llevar a cabo tareas encomendadas en el tiempo previsto, trabajando con las fuentes de información indicadas, de acuerdo con las pautas marcadas por el profesorado.
8. APRENDIZAJE AUTÓNOMO - Nivel 2: Llevar a cabo las tareas encomendadas a partir de las orientaciones básicas dadas por el profesorado, decidiendo el tiempo que se necesita emplear para cada tarea, incluyendo aportaciones personales y ampliando las fuentes de información indicadas.
9. APRENDIZAJE AUTÓNOMO: Detectar deficiencias en el propio conocimiento y superarlas mediante la reflexión crítica y la elección de la mejor actuación para ampliar este conocimiento.
10. COMUNICACIÓN EFICAZ ORAL Y ESCRITA - Nivel 1: Planificar la comunicación oral, responder de manera adecuada a las cuestiones formuladas y redactar textos de nivel básico con corrección ortográfica y gramatical.
11. COMUNICACIÓN EFICAZ ORAL Y ESCRITA - Nivel 2: Utilizar estrategias para preparar y llevar a cabo las presentaciones orales y redactar textos y documentos con un contenido coherente, una estructura y un estilo adecuados y un buen nivel ortográfico y gramatical.
12. COMUNICACIÓN EFICAZ ORAL Y ESCRITA: Comunicarse de forma oral y escrita con otras personas sobre los resultados del aprendizaje, de la elaboración del pensamiento y de la toma de decisiones; participar en debates sobre temas de la propia especialidad.
13. TERCERA LENGUA: Conocer una tercera lengua, que será preferentemente inglés, con un nivel adecuado de forma oral y por escrito y en consonancia con las necesidades que tendrán las tituladas y los titulados en cada enseñanza.
14. TRABAJO EN EQUIPO - Nivel 1: Participar en el trabajo en equipo y colaborar, una vez identificados los objetivos y las responsabilidades colectivas e individuales, y decidir conjuntamente la estrategia que se debe seguir.
15. TRABAJO EN EQUIPO - Nivel 2: Contribuir a consolidar el equipo planificando objetivos, trabajando con eficacia y favoreciendo la comunicación, la distribución de tareas y la cohesión.
16. TRABAJO EN EQUIPO: Ser capaz de trabajar como miembro de un equipo interdisciplinar ya sea como un miembro más, o realizando tareas de dirección con la finalidad de contribuir a desarrollar proyectos con pragmatismo y sentido de la responsabilidad, asumiendo compromisos teniendo en cuenta los recursos disponibles.

METODOLOGÍAS DOCENTES

- Un caso de estudio que vehicula todo el curso.
- Material en forma de transparencias, para el estudio y repaso de los contenidos.
- Un proyecto de desarrollo de software en grupos de seis a ocho personas como mecanismo fundamental de aprendizaje y evaluación.
- Sesiones de tutoría del trabajo realizado en las clases de laboratorio.

OBJETIVOS DE APRENDIZAJE DE LA ASIGNATURA

- Profundizar en el concepto y la importancia de la ingeniería del software.
- Conocer y aplicar un enfoque nuevo (las metodologías ágiles) para el desarrollo de software.
- Ser capaces de llevar a cabo un desarrollo completo de tamaño medio en un entorno similar al profesional: exceso de requisitos, falta de tiempo, trabajo en equipo, retro-alimentación continua.
- Reconocer la importancia de las pruebas y la documentación en el desarrollo de software.
- Aprender a usar algunas herramientas básicas de soporte al desarrollo de software

HORAS TOTALES DE DEDICACIÓN DEL ESTUDIANTADO

Tipo	Horas	Porcentaje
Horas grupo pequeño	30,0	20.00
Horas aprendizaje autónomo	90,0	60.00
Horas grupo grande	30,0	20.00

Dedicación total: 150 h

CONTENIDOS

Introducción a las metodologías ágiles de desarrollo de software.

Descripción:
Presenta los conceptos básicos de la metodología ágil que se van a usar a lo largo del curso

Objetivos específicos:
- Repasar los conceptos básicos de la ingeniería del software (IS) que aplican al curso
- Introducir los conceptos básicos de las metodologías ágiles de desarrollo de software, mostrando las diferencias con la metodología dirigida por un plan estudiada en INEP
- Analizar el impacto de estos cambios en las distintas etapas del ciclo de vida del software

Actividades vinculadas:
Examen Parcial (ExParcial), Examen Final (ExFinal)

Dedicación: 2h
Grupo grande/Teoría: 2h

Conceptos básicos de las metodologías ágiles. La metodología SCRUM

Descripción:
Conocer y aplicar los conceptos básicos de las metodologías ágiles en general, y Scrum en particular

Objetivos específicos:
- Conocer el concepto y tipología de backlogs en las metodologías ágiles
- Conocer el concepto de historia de usuario y similares (épica, …) y su relación con el concepto clásico de requisito
- Aprender las propiedades de las historias de usuario, así como las técnicas para su priorización
- Introducir los principios básicos de Scrum
- Introducir los roles en los proyectos Scrum
- Presentar los eventos clave de la ceremonia ágil en Scrum

Actividades vinculadas:
Examen Parcial (ExParcial), Examen Final (ExFinal)

Dedicación: 15h
Grupo grande/Teoría: 10h
Aprendizaje autónomo: 5h

Técnicas avanzadas de apoyo al desarrollo de software

Descripción:
Aprender algunas técnicas que pueden acelerar el proceso de desarrollo de software y mejorar su calidad

Objetivos específicos:
- Comprender los conceptos de integración continua y despliegue continuo y motivar su importancia en el proceso de desarrollo de software
- Mostrar un par de herramientas de apoyo a estas actividades
- Discutir las limitaciones y desafíos inherentes a la adopción de estas herramientas de automatización
- Presentar algunas herramientas básicas de apoyo a la gestión de la calidad del software
- Aprender a hacer un buen uso de las herramientas generativas de código

Actividades vinculadas:
Examen Final (ExFinal)

Dedicación: 10h
Grupo grande/Teoría: 6h
Aprendizaje autónomo: 4h

Pruebas del software. Documentación

Descripción:
Comprender y aplicar las técnicas básicas de pruebas y documentación de código

Objetivos específicos:
- Motivar y justificar la necesidad de las pruebas del software
- Introducir los distintos tipos de pruebas que existen
- Visualizar la relación entre las pruebas del software y las demás etapas de la ingeniería del software, en el contexto de las metodologías ágiles. Definition of Done
- Reconocer las buenas prácticas asociadas a la calidad del diseño y del código
- Dar unas pautas básicas para la documentación del código

Actividades vinculadas:
Examen Parcial (ExParcial), Examen Final (ExFinal)

Dedicación: 14h
Grupo grande/Teoría: 8h
Aprendizaje autónomo: 6h

Conclusiones

Descripción:
Realizar un compendio de los conocimientos y prácticas impartidos en la asignatura y avanza los desafíos que se estudiarán en la asignatura AMEP

Objetivos específicos:
- Resumir y relacionar los conocimientos y prácticas impartidas en la asignatura
- Enumerar las limitaciones de las técnicas vistas
- Avanzar algunas soluciones a estas limitaciones, y posibles técnicas alternativas
- Dar una panorámica de métodos avanzados no presentados en la asignatura

Actividades vinculadas:
Examen Final (ExFinal)

Dedicación: 2h
Grupo grande/Teoría: 2h

Desarrollo del proyecto

Descripción:
Aplicar los conocimientos introducidos durante el curso en un proyecto de laboratorio desarrollado en equipo

Objetivos específicos:
- Aprender a usar eficazmente un conjunto de herramientas de soporte al desarrollo de software adecuado para los objetivos del curso
- Aprender a elaborar un sistema de software de forma incremental y con un ciclo de vida por iteraciones
- Aplicar las nociones teóricas introducidas en la clase de teoría
- Afrontar los retos típicos de programación de un sistema aplicando los principios y métodos de la ingeniería del software con un enfoque ágil

Actividades vinculadas:
Proyecto (Proj). Examen Parcial (ExParcial), Examen Final (ExFinal)

Dedicación: 105h
Grupo pequeño/Laboratorio: 30h
Aprendizaje autónomo: 75h

SISTEMA DE CALIFICACIÓN

La nota final de la asignatura se obtiene mediante la siguiente fórmula:

Nota Final = 0.1*ExParcial+0.2*ExFinal+0.7*Proj,

donde:
* ExParcial y ExFinal son las notas obtenidas en las dos pruebas correspondientes. Ambos exámenes combinarán preguntas puramente teóricas y preguntas relacionadas con el desarrollo del proyecto. En este segundo caso se tratará de resolver, en un tiempo limitado y de forma individual, alguno de los aspectos que, de forma más compleja o con algún matiz de diferencia, ya se han resuelto durante la realización en equipo del proyecto.
* Proj es la nota del proyecto. El proyecto se organizará en cuatro iteraciones, con una entrega por cada una. La primera iteración corresponde a la incepción del problema, y las otras tres se corresponden a diversas etapas en el desarrollo del software. Los profesores corregirán y puntuarán las entregas y darán retroalimentación a los estudiantes. Las entregas se podrán complementar con un pequeño cuestionario a resolver en la clase siguiente a la entraga. Los profesores pueden decidir asignar notas distintas a los integrantes de un mismo equipo de desarrollo del proyecto en base a diversas fuentes de información (informes asociados a las entregas, trazas en las herramientas de desarrollo usadas, respuestas a los cuestionarios, participación en la dinámica de clase, etc.).).

No Presentado: se evaluará como No Presentado todo estudiante que no se presente al examen final y que no figure como integrante del equipo en la entrega final.

También habrá una prueba de reevaluación para aquellos estudiantes con una nota final menor que 5 y mayor que 2. La prueba de reevaluación constará de dos partes, una para cada examen. Se podrán presentar a reevaluar la primera parte sólo aquellos estudiantes con ExParcial < 5. Se podrán presentar a reevaluar la segunda parte sólo aquellos estudiantes con ExFinal < 5. La pràctica no se puede reevaluar.

NORMAS PARA LA REALIZACIÓN DE LAS PRUEBAS.

Para aprobar el proyecto, los estudiantes deben asistir regularmente a las clases de laboratorio salvo causas justificadas, y durante el curso deben dejar patente su contribución individual al desarrollo del proyecto.

BIBLIOGRAFÍA

Básica:
- Pressman, Roger S. Ingeniería del software : un enfoque práctico [en línea]. 9a ed. México [etc.]: McGraw Hill, 2021 [Consulta: 16/02/2024]. Disponible a: https://www-ingebook-com.recursos.biblioteca.upc.edu/ib/NPcd/IB_BooksVis?cod_primaria=1000187&codigo_libro=4272. ISBN 9781456287726.
- Larman, Craig. Applying UML and patterns : an introduction to object-oriented analysis and design and iterative development. 3th ed. Upper Saddle River, N.J: Prentice Hall PTR, 2005. ISBN 0131489062.
- Cohn, Mike. User stories applied : for agile software development [en línea]. Boston, MA: Addison-Wesley, 2004 [Consulta: 01/03/2024]. Disponible a: https://ebookcentral-proquest-com.recursos.biblioteca.upc.edu/lib/upcatalunya-ebooks/detail.action?pq-origsite=primo&docID=7116223. ISBN 9780321205681.
- Cohn, Mike. Agile estimating and planning [en línea]. Upper Saddle River, NJ [etc.]: Prentice Hall Professional Technical Reference, 2006 [Consulta: 05/03/2024]. Disponible a: https://ebookcentral-proquest-com.recursos.biblioteca.upc.edu/lib/upcatalunya-ebooks/detail.action?pq-origsite=primo&docID=7114929. ISBN 0131479415.
- Rasmusson, Jonathan. The Agile samurai : how agile masters deliver great software [en línea]. Raleigh ; Dallas: The Pragmatic Bookshelf, 2010 [Consulta: 28/01/2025]. Disponible a: https://ebookcentral.proquest.com/lib/upcatalunya-ebooks/detail.action?pq-origsite=primo&docID=5307650. ISBN 9781934356586.

RECURSOS

Otros recursos:
- Guía de Scrum: https://www.scrum.org/resources/scrum-guide />

- Agile Manifesto: http://agilemanifesto.org/