Universitat Autònoma de Barcelona

2022/2023

Gestión del Desarrollo de Software

Código: 102789
Créditos ECTS: 3

Titulación	Tipo	Curso	Semestre
2502441 Ingeniería Informática	OB	3	2
2502441 Ingeniería Informática	OT	4	2

Contacto

Nombre: Marc Talló Sendra
Correo electrónico: marc.tallo@uab.cat

Uso de idiomas

Lengua vehicular mayoritaria: catalán (cat)
Algún grupo íntegramente en inglés: No
Algún grupo íntegramente en catalán: Sí
Algún grupo íntegramente en español: No

Prerequisitos

No hay ningún prerrequisito previo para cursar esta asignatura. Pero se aconseja cursar también las asignaturas:

- Arquitectura y tecnología del software
- Laboratorio integrado del software

Objetivos y contextualización

En la asignatura de ingeniería del software de segundo se ha realizado una visión global del proceso de ingeniería del software. Se Han aprendido los conceptos básicos y, para aquellos que hayan cursado las asignaturas del primer semestre de la mención de ingeniería del software, han tenido que profundizar en algunos de estos conceptos como el análisis de requisitos, el diseño y las técnicas de test y control de calidad.

En esta asignatura aprenderás a gestionar el cambio. Todo proceso de desarrollo de software está sometido a cambios de todo tipo. Cambios provocados por la necesidad de mejora del producto desarrollado. Cambios en los requerimientos iniciales. Cambios en los plazos o el equipo de trabajo, presupuesto, etc. En definitiva, al final de este curso deberá ser capaz de gestionar cualquier cambio que suponga una desviación de la planificación inicial.

Más concretamente, los objetivos de este curso son:

- Profundizar en las técnicas de control y desarrollo del software.
- Profundizar en el conocimiento de las técnicas de administración necesarias para planificar, monitorizar y controlar los proyectos de software.
- Aprender a utilizar algunas de las herramientas existentes para llevar a cabo el control necesario del desarrollo de software.

Finalmente, para el correcto seguimiento de este curso, el equipo docente asume que todos los estudiantes han alcanzado las competencias trabajadas en ingeniería del software.

Competencias

Ingeniería Informática
- Adquirir hábitos de pensamiento.
- Adquirir hábitos de trabajo personal.
- Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles.
- Capacidad para diseñar, desarrollar, evaluar y asegurar la accesibilidad, ergonomía, usabilidad y seguridad de los sistemas, servicios y aplicaciones informáticas, así como de la información que gestionan.
- 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.

Resultados de aprendizaje

1. Definir y gestionar la documentación que se genera durante el desarrollo de una aplicación software.
2. Desarrollar la capacidad de análisis, síntesis y prospectiva.
3. Desarrollar un pensamiento y un razonamiento crítico.
4. Planificar la integración de las diferentes componentes desarrolladas en el proceso de codificación.
5. Resolver problemas de integración del software con iniciativa y autonomía.
6. Trabajar de forma autónoma.

Contenido

BLOQUE 1: Gestión de Proyectos (GP)

1. Introducción. Conceptos de la GP
Tareas y actividades. Productos, paquetes y roles. Modelo de tareas. Matriz de perfiles. Estructuras de organizaciones: tipos y visualización.

2. Actividades de GP
Planificación, organización, control y finalización

3. Metodologías de GP
Metodología de Royce. Desarrollo iterativo e incremental. Desarrollo ágil.

BLOQUE 2: Gestión de la Configuración (GC)

1. Introducción. Conceptos de la GC
Ítems de configuración y agregados de GC. Versiones y configuraciones. Peticiones de cambios. Promociones y releases. Depósitos y espacios de trabajo. Esquemas de identificación de versiones. Herramientas de GC.

2. Actividades de la GC
Ítems de configuración e identificación de los agregados de la GC. Gestión de la promoción. Gestión de los releases. Gestión de las ramas. Gestión de las variantes. Gestión del cambio.

3. Gestionando la GC
Documentar la GC. Asignación de responsabilidades en la GC. Organizar las actividades de la GC.

Metodología

El proceso de aprendizaje del alumno se basará en estos dos tipos de actividades:

Teoría y Problemas. Con el planteamiento de casos de estudio, los alumnos practicarán en clase los conceptos teóricos que se habrán introducido al inicio de la clase mediante soporte documental y pizarra. A continuación, los alumnos trabajarán en pequeños grupos y discutirán entre ellos las posibles soluciones hasta concluir y elegir una de ellas. En el tramo final de la clase, se pondrán en común todas las soluciones planteadas.

Prácticas. En las clases de prácticas se trabajará en grupos de dos personas y se practicará el uso de las herramientas de gestión de la configuración necesarias en todo proyecto de desarrollo de software.

Toda la documentación y material necesario para el seguimiento del curso se encontrará disponible en el Campus Virtual.

Competencias transversales: Las competencias transversales serán trabajadas y evaluadas en varios momentos a lo largo del curso. Algunos ejemplos de cómo se trabajarán son los siguientes:

- T01.01 - Desarrollar un modo de pensamiento y razonamiento críticos:
  En las clases de teoría se explican ejemplos de los problemas que aparecen en la gestión del desarrollo de software. Se explican las soluciones que históricamente se han aplicado y se valoran los pros, contras y errores.
  En la práctica el alumnado debe conseguir construir un modelo de desarrollo del software dentro de un proyecto y debe ser capaz de encontrar una estrategia para gestionarlo.
  En el examen de teoría hay preguntas que obligan al alumno a hacer razonamientos sobre la materia para encontrar la respuesta.
- T02.03 - Gestionar el tiempo y los recursos disponibles. Trabajar de manera organizada:
  Esta asignatura trata sobre cómo gestionar el tiempo, recursos y organizar el trabajo dentro de un proyecto de desarrollo de software.
  En la segunda práctica del alumnado debe especificar los recursos, la planificación temporal y cuál será la metodología utilizada de un proyecto de software.
- T02.04 - Prevenir y solucionar problemas:
  Un de los temas de la asignatura trata sobre cómo controlar un proyecto y prevenir sus riesgos.
  En la primera práctica del alumnado ha de diseñar un repositorio pensando en los problemas que pueden surgir durante el desarrollo de un software concreto.
  En la segunda práctica del alumnado tiene que gestionar un proyecto incluyendo los posibles cambios de las especificaciones que se producirán durante el desarrollo.
- T02.05 - Tomar decisiones propias:
  En la práctica, el alumnado debe tomar decisiones sobre cómo organiza un repositorio de software y cómo organiza un proyecto.

NOTA: La metodología docente y la evaluación propuestas pueden experimentar alguna modificación en función de las restricciones a la presencialidad que impongan las autoridades sanitarias.

Nota: se reservarán 15 minutos de una clase dentro del calendario establecido por el centro o por la titulación para que el alumnado rellene las encuestas de evaluación de la actuación del profesorado y de evaluación de la asignatura o módulo.

Actividades

Título	Horas	ECTS	Resultados de aprendizaje
Tipo: Dirigidas			
Clase de Problemas	13	0,52	5
Clase de Teoría	13	0,52	1, 2, 3, 4, 5
Sesiones de Prácticas	6	0,24	1, 2, 3, 4
Tipo: Supervisadas			
Tutorías	9	0,36	1, 4, 5
Tipo: Autónomas			
Preparación de ejercicios y resolución de supuestos teóricos	15	0,6	3, 5, 6

Preparación y realización de los ejercicios prácticos	15	0,6	1, 4, 5, 6	

Evaluación

La asignatura consta de 2 tipos de evaluaciones: teoría (NTeo) y prácticas (NLAB). El peso de cada una de estas evaluaciones es el siguiente:

NFinal = 0.5 * NTeo + 0.5 * NLAB

NLAB = 0.5 * NLab1 (SPMP) + 0.5 * NLab2

Para calcular la nota NFinal necesario que la nota de cada una de las diferentes evaluaciones (NTeo y NLAB) sean todas y cada una de ellas igual o superior a 5. En caso contrario, la asignatura está suspendida y la calificación final de la asignatura será la nota mínima de NTeo y NLAB.

Si NTeo no es superior a 5, el alumno se podrá volver a evaluar de NTeo durante las pruebas de recuperación.

Para calcular la nota NLAB, todas y cada una de las notas NLAB (i), es decir, todas y cada una de las entregas de prácticas, deberán ser superiores a 4. En caso contrario, NLAB es la nota mínima de las notas NLAB (i). Las evaluaciones mediante casos prácticos y la redacción de informes de prácticas forman parte del bloque de evaluación de prácticas (NLAB). Las fechas de entrega y la forma de entrega de los diferentes casos prácticos estarán consignados en los mismos enunciados. No hay segunda convocatoria para la nota NLAB ni por ninguna de las notas NLAB (i), es decir, no hay segunda convocatoria para las sesiones de laboratorio.

Para cada actividad de evaluación, se indicará un lugar, fecha y hora de revisión en la que el estudiante podrá revisar la actividad con el profesor. En este contexto, se podrán hacer reclamaciones sobre la nota de la actividad, que serán evaluadas por el profesorado responsable de la asignatura. Si el estudiante no se presenta en esta revisión, no se revisará posteriormente esta actividad.

Un estudiante se considera No Evaluable únicamente si las actividades de evaluación que ha realizado están aprobadas (pero no ha hecho todaslas actividades de evaluación del curso). Si una actividad de evaluación está suspendida, la asignatura está suspendida (independientemente del número de actividades a las que se haya presentado).

La matrícula de honor se puede conseguir con una nota media superior o igual a 9,0. Debido a que hay un número limitado de matrículas de honor que se pueden dar por grupo, se otorgarán por orden de nota de mayor a menor.

Sólo se convalidará la nota de prácticas (NLAB) de años anteriores.

Siempre se informará al Campus Virtual sobre posibles cambios en el desarrollo de la asignatura ya que se entiende que es el mecanismo habitual de intercambio de información entre profesor y estudiantes.

Sin perjuicio de otras medidas disciplinarias que se estimen oportunas, y de acuerdo con la normativa académica vigente, las irregularidades cometidas por un estudiante que puedan conducir a una variación de la calificación se calificarán con un cero (0). Por ejemplo, plagiar, copiar, dejar copiar, ..., implicará suspender esta actividad de evaluación con un cero (0). Las actividades de evaluación calificadas de esta forma y por este procedimiento no serán recuperables. Si es necesario superar cualquiera de estas actividades de evaluación para aprobar la asignatura, esta asignatura quedará suspendida directamente, sin oportunidad de recuperarla en el mismo curso.

Actividades de evaluación

Título	Peso	Horas	ECTS	Resultados de aprendizaje
Examen de Teoría	50%	3	0,12	1, 2, 3, 4, 5, 6

Informes / trabajos de prácticas	50%	1	0,04	1, 2, 3, 4, 5, 6

Bibliografía

- Software war stories : case studies in software management / Donald J. Reifer. Reifer, Donald J., author.; Rutkowski, Michael. 2013 . Biblioteca UAB
- Agile management for software engineering : applying the theory of constraints for business results / David J. Anderson . Anderson, David J. (David James), 1939 cop. 2004 . Biblioteca UAB
- Software configuration management patterns : effective teamwork, practical integration / Stephen P. Berczuk with Brad Appleton . Berczuk, Stephen P.; Appleton, Brad . cop. 2003 . Biblioteca UAB
- Bruegge, B. I Dutoit A.H. - Object-Oriented Software Engineering. Pearson, Prentice-Hall, 2004.
- Berczuk, S. and Appleton B.Software - Configuration Management Patterns: Effective Teamwork, Practical Integration. Addison-Wesley, 2002.
- Steve McConnell - Rapid Development: Taming Wild Software Schedules (Developer Best Practices) - Microsoft Press Nº 1 Ed. 1996.

Software

GIT:

Git es un sistema de control de versiones distribuido de código abierto y gratuito diseñado para manejar todo, desde proyectos pequeños a muy grandes, con velocidad y eficiencia.

https://git-scm.com/

BITBUCKET:

Bitbucket es más que solo administración de código Git. Bitbucket ofrece a los equipos un lugar para planificar proyectos, colaborar en el código, probar e implementar.

https://bitbucket.org/product/