TecnoCampus Escola Superior Politècnica
Centre adscrit a la upf. Universitat Pompeu Fabra Barcelona

DOBLE GRADO EN INFORMÁTICA DE GESTIÓN Y SISTEMAS DE INFORMACIÓN/GRADO EN DISEÑO Y PRODUCCIÓN DE VIDEOJUEGOS

107212 - INGENIERÍA DEL SOFTWARE I

Información general

- Tipo de asignatura : Obligatoria
- Coordinador : Adso Fernández Baena
- Curso: Segundo
- Trimestre: Primero
- Créditos: 4
- Profesorado:
  - Eugeni Fernández González <efernandezgo@tecnocampus.cat>
  - Eduard De Bru De Sala Castells <debru@tecnocampus.cat>

Idiomas de impartición

- Catalán

Competencias que se trabajan

Básica

- B2_Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio
- B5_Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.

Común

- CIN1_Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y la legislación y normativa vigente.
- CIN2_Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social.
- CIN3_Capacidad para comprender la importancia de la negociación, los hábitos de trabajo efectivos, el liderazgo y las habilidades de comunicación en todos los entornos de desarrollo de software
- CIN4_Capacidad para elaborar el pliego de condiciones técnicas de una instalación informática que cumpla los estándares y normativas vigentes
- CIN5_Conocimiento , administración y mantenimiento sistemas , servicios y aplicaciones informáticas
- CIN8_Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.
- CIN13_Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los sistemas de información, incluidos los basados en web.
- CIN16_Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.

Específica

- EFB4_Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.
- EIS1_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.
- EIS4_Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.

Transversal

- T1_Que los estudiantes conozcan un tercer idioma, que será preferentemente inglés, con un nivel adecuado de forma oral y por escrito y de acuerdo con las necesidades que tendrán las graduadas y los graduados en cada titulación
- T2_Que los estudiantes tengan capacidad para trabajar como miembro de un equipo interdisciplinario 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

Descripción

La asignatura de Ingeniería del Software 1 del primer trimestre de segundo curso, es la primera de las tres asignaturas denominadas Ingeniería del Software. Su impartición está pensada en dedicar 3 ECTS en la parte de teoría y 1 ECTS a practicar los conceptos expuestos en teoría.

Esta asignatura será la que introducirá el concepto de Ingeniería del Software, haciendo énfasis en la disciplina del análisis, la especificación de requerimientos y una introducción a la modelización de lo que llamamos el Dominio del software.

Esta asignatura dispone de recursos metodológicos y digitales para hacer posible su continuidad en modalidad no presencial en el caso de ser necesario por motivos relacionados con el Covid-19.

De esta forma se asegurará la consecución de los mismos conocimientos y competecias que se especifican en este plan docente.

Resultados de aprendizaje

- Utilizar de forma apropiada teorías, procedimientos y herramientas en el desarrollo profesional de la ingeniería informática en todos sus ámbitos (especificación, diseño, implementación, despliegue -implantación- y evaluación de productos) de forma que se demuestre la comprensión de los compromisos adoptados en las decisiones de diseño.
- Demostrar conocimiento de la dimensión ética en la empresa: la responsabilidad social y corporativa en general y, en particular, las responsabilidades civiles y profesionales del ingeniero en informática.
- Usar las herramientas de un entorno de desarrollo de software para crear y desarrollar aplicaciones.
- Demostrar conocimiento y saber aplicar las técnicas apropiadas para modelar y analizar los diferentes tipos de decisiones.
- Controlar la calidad y diseñar pruebas en la producción de software.
- Definir y gestionar los requisitos de un sistema software.

A un nivel más concreto, al finalizar la asignatura los estudiantes deben ser capaces de:

- RA1 - Entender las etapas que requiere la construcción de software partiendo de la observación de la realidad (incluido lo que pide el usuario)
- RA2 - Entender perfectamente porque es necesario dividir la complejidad para abordar cualquier proyecto de Ingeniería de Software.
- RA3 - Entender los requisitos de un sistema de software.
- RA4 - Reconocer las propiedades deseables de las especificaciones.
- RA5 - Hacer el análisis del sistema de forma correcta.
- RA6 - Modelar el sistema con la notación UML.
- RA7 - Crear el modelo del comportamiento de un sistema en notación UML.
- RA8 - Transformar el modelo de análisis en un modelo de diseño en notación UML.
- RA9 - Aplicar patrones de diseño básicos (GRASP).

Metodología de trabajo

Todos los conceptos teóricos de la materia se tratarán en las clases de teoría (grupos grandes) de la asignatura. En estas clases se introducen los conceptos básicos del análisis y diseño del software mostrando su aplicación con ejercicios resueltos por el docente. Se recomienda que antes de cada sesión los estudiantes se lean el material publicado en la plataforma virtual. En las clases se pedirá la participación de los estudiantes de manera individual o en grupo, para resolver diferentes problemas propuestos con o sin anticipación. Estas actividades, que por su naturaleza de optatividad y brevedad no aparecen reflejadas en este documento, servirán al estudiante como instrumento de autoevaluación del logro de los contenidos de la materia y podrán ser utilizados por parte del docente para tomar decisiones sobre la calificación final del estudiante pero nunca en detrimento de la calificación numérica calculada según el sistema de calificación antes indicado.

Los conceptos de carácter más práctico serán trabajados en grupos pequeños (de laboratorio) donde se presentan trabajos de complejidad media, que requieren la aplicación de los conocimientos adquiridos en las clases más teóricas. En estas sesiones se darán las herramientas adecuadas para resolver las

actividades programadas pero se espera que estas alarguen desde el punto de vista temporal, más allá de las horas de laboratorio y que, en consecuencia, los estudiantes deban finalizar durante el tiempo de aprendizaje autónomo.

Normas de realización de las actividades
Para cada actividad, los docentes informarán de las normas y condiciones particulares que las rijan.
Las actividades unipersonales presuponen el compromiso del estudiante de realizarlas de manera individual. Se considerarán suspendidas todas aquellas actividades en las que el estudiante no se ajuste a este compromiso, independientemente de su papel (origen o destino).
Igualmente, las actividades que se deban realizar en grupos presuponen el compromiso por parte de los estudiantes que lo integran de realizarlas en el seno del grupo. Se considerarán suspendidas todas aquellas actividades en que el grupo no haya respetado este compromiso con independencia de su papel (origen o destino).
En las actividades realizadas en grupo el docente puede, en base a la información de que disponga, personalizar la calificación para cada integrante del grupo.
Cualquier actividad no entregada se considerará puntuada con cero puntos.
Es potestad de los docentes aceptar o no entregas fuera de los plazos que se indiquen. En caso de que estas entregas fuera de plazo se acepten, es potestad del docente decidir si aplica alguna penalización y la cuantía de la misma.

Este curso, a causa de la situación generada por el COVID, algunas de las sesiones de gran grupo se harán en formato híbrido: presencial y en línea (via streaming).
Esto permitirá que los estudiantes puedan asistir de forma rotativa a las clases presenciales, respetando el máximo de estudiantes por aula que establecen las medidas de distancia social.
Cuando no les toque sesión presencial podrán seguir la clase en línea desde casa.

Con respecto a las sesiones prácticas en espacios mas reducidos (como laboratorios), si hiciese falta se trabajaráde forma simultanea en diversos espacios con el objetivo de garantizar que se cumplan las condiciones establecidas por los protocolos de seguridad.

Contenidos

1. Especificación y requerimientos del software
   1.1 Especificación y alcance de la aplicación.
   1.2 Definición, calidad y tipos de requerimientos.
   1.3 Estudio de los Casos de uso.

2. Modelo del dominio
   2.1 De los casos de uso en el modelo del dominio.
   2.2 Clases, asociaciones y atributos.
   2.3 Agregación y composición.
   2.4 Clase asociativa.
   2.5 Jerarquía de clases.
   2.6 Guías de modelado.

3. Modelo de diseño
   3.1 Del modelo del dominio al modelo de diseño.
   3.2 Modelo de comportamiento: diagrama de interacción y secuencia.
   3.3 Patrones de asignación de responsabilidades (GRASP).
   3.4 Diagrama de clases de diseño.

4. Modelo de Implementación
   4.1 Del diseño a la implementación.
   4.2 Definición de las clases a partir del diagrama de clases de diseño.
   4.3 Sentencias de métodos a partir de los diagramas de interacción.
   4.4 Clases contenedoras
   4.5 Orden de implementación.

5. Introducción a la Ingeniería del Software
   5.1 ¿Qué es la ingeniería del software.
   5.2 Características particulares del software.
   5.3 Proceso Software en cascada.
   5.4 Proceso Software basado en prototipos.
   5.5 Proceso Software Iterativo.

Actividades de aprendizaje

CRITERIO GENERAL DE EVALUACIÓN

Para superar las actividades evaluativas que se proponen a continuación, los estudiantes deberán demostrar

- Que han adquirido los conocimientos teóricos relativos a los contenidos de la asignatura y que su comprensión les permite llevarlos a la práctica [MECES-2 punto a, punto c]
- Que tienen la capacidad de recopilar e interpretar los requerimientos de un sistema sobre las que desarrollar los modelos necesarios [MECES-2 punto c]
- Que pueden desarrollar soluciones a problemas que, si bien son similares a otros vistos anteriormente, presentan aspectos que son nuevos [MECES-2 punto f]

PONDERACIÓN

El conjunto de las prácticas representan un 40% de la nota final, pero la evaluación de las mismas se hace en función del nivel final alcanzado por el alumno.

PRÁCTICAS

Práctica 1

- Cada grupo tiene asignado un vídeo donde se explican un conjunto de requerimientos muy básicos.
- Los alumnos deben se capaces de entender los "límites" de lo que se está pidiendo y el "contexto" que rodea el que se le esta pidiendo.
- Objetivos específicos: Al finalizar la actividad los estudiantes deben ser capaces de redactar un documento que explique

Que se pide.

- Que debe "estar funcionando" para poder hacer lo que se pide.
- Una estimación de tiempo en horas de lo que traerá el trabajo (sin tener ninguna idea de cómo estimar).

La práctica 1 dará evidencia de los resultados de aprendizaje: RA1, RA3, RA4

Práctica 2

- Se trata de analizar el mismo enunciado de la práctica 1 pero esta vez el resultado del análisis debe ser más cuidadoso ..

Que se pide:

- Especifica cuál es el alcance de la aplicación.
- Enunciar "todos" los requerimientos que se implementarán.
- Enunciar "todos" los requerimientos que aunque no se tengan que implementar, son necesarios.
- Dibujar el caso de uso que pide el documento de la práctica 1.

La práctica 2 dará evidencia de los resultados de aprendizaje: RA2, RA3, RA4, RA5

Práctica 3

- Se trata de evolucionar el proyecto que se ha empezado con la práctica 2 para llegar hasta el modelo de diseño que se debe programar.

Que se pide:

- Entregar un diagrama con las estructuras conceptuales principales del sistema.
- Enumerar cuáles serán las clases principales que se deben implementar en el sistema.
- Hacer el diagrama de clases básicas del sistema.
- Hacer uno de los diagrama de interacción necesarios.
- Hacer uno de los diagramas de secuencia necesarios.

La práctica 3 dará evidencia de los resultados de aprendizaje: RA6, RA7, RA8

Práctica 4

- Se trata de evolucionar el proyecto que se ha empezado con la práctica 2 para llegar hasta el punto de la codificación.

Que se pide:

- Diagrama de clases que refleje los patrones GRASP que se aplicarán al diseño creado por la práctica 3.
- Implementación de las clases presentadas en el diagrama del punto previo.
- Cálculo de la complejidad ciclomática del método más largo de todos los que se hayan implentat.

La práctica 4 dará evidencia de los resultados de aprendizaje: RA6, RA7, RA8, RA9

Las prácticas 1,2,3 y 4 1 tienen que ver con las siguientes competencias comunes y específicas (entre paréntesis los aspectos más relevantes de cada competencia a los que la asignatura contribuye)

- T2 (tengan capacidad para trabajar como miembro de un equipo)
- B5 (Desarrollar su grado de autonomía)
- CIN3 (Comprender la importancia de los hábitos de trabajo efectivos)
- B2 (Aplicar conocimientos a su trabajo)
- CIN13 (Conocimiento y aplicación de herramientas)

PRUEBAS

Prueba 1 : Bloques 1, 2 y 3

- Prueba individual de los conceptos teóricos y procedimientos prácticos de los tres primeros bloques de la asignatura.
- Esta prueba representa el 30% de la calificación final de la asignatura.
- Objetivos específicos: El objetivo de esta actividad es evaluar si el estudiante:
    - Entiende los conceptos de la Ingeniería de de Software.
    - Puede explicar "que se" un proceso de software
    - Entiende perfectamente porque hay que modelar como base de la Ingeniería de de Software.
    - Sabe dividir la complejidad de un problema pequeño que se le plantee.
    - Entiende que es el modelo de dominio.
    - Entiende las rel·lacions básicas entre clases.

La prueba 1 dará evidencia de los resultados de aprendizaje: RA1, RA2, RA3, RA4, RA5

Prueba 2 : bloques 4 y 5

- Prueba individual de los conceptos teóricos y procedimientos prácticos de los bloques 4 y 5 de la asignatura.
- Esta prueba representa el 30% de la calificación final de la asignatura.
- Objetivos específicos: El objetivo de esta actividad es evaluar si el estudiante:
    - Sabe transformar un modelo de especificación en UML a un modelo de diseño.

- Sabe crear un diagrama de clases.
- Sabe crear un diagrama de interacción de clases.
- Sabe crear un diagrama de secuencia de clases.
- Sabe aplicar los patrones GRASP básicos.
- Sabe codificar una clase en base a un diagrama de clases.

La prueba 2 dará evidencia de los resultados de aprendizaje: RA5, RA6, RA7, RA8, RA9, CIN2 (Planificar y concebir proyectos)

Las pruebas 1 y 2 tienen que ver con las siguientes competencias comunes y específicas (entre paréntesis los aspectos más relevantes de cada competencia a los que la asignatura contribuye)

- C1N (Diseña y desarrollar aplicaciones)
- CIN2 (Planificar i concebir proyectos)
- CIN4 (Elaborar un pliego de condiciones técnicas)
- CIN5 (Conocimiento de aplicaciones informáticas)
- CIN8 (Capacidad para analizar y diseñar aplicaciones de forma robusta)
- CIN16 (Conocimiento y aplicación de metodologías)
- EIS1 (Desarrollar sistemas software que satisfagan los requisitos del usuario)
- EIS4
    - (PARTE1: Identificar i analizar problemas)
    - (PARTE2: diseñar , desarrollar, implantar y verificar soluciones de software)
- EFB4 (Conocimientos básicos sobre el uso y la programación de ordenadores)
- C1N (Diseñar y evolucionar aplicaciones)
- T1 (conozcan una tercera lengua)

Lectura y Comprensión (aprendizaje autónomo)

- Lectura y comprensión de capítulos escogidos por el profesor de los libros de la bibliografía y material de clase.
- Material de apoyo: Libros (disponibles en la biblioteca) y material del curso.
- Habrá responder a cuestionario sobre la lectura.

Objetivos específicos:

- Entender y aplicar conceptos complejos de ingeniería del software a partir de la lectura y estudio del material propuesto por el profesor.

Sistema de evaluación

La nota final se calculará con las calificaciones de las actividades ponderadas de la forma siguiente:
· Prueba 1: 30%
· Prueba 2: 30%
· Prácticas de la 1 a la 5: 40% (8% cada una de las prácticas)

Con las ponderaciones anteriores, las prácticas de laboratorio tienen un peso del 40% y las pruebas un 60%.

Sólo podrán recuperarse las pruebas 1 y 2 en una única prueba de toda la asignatura (**las prácticas no se podrán recuperar**). El 60% de la nota final de la asignatura será la mayor entre la prueba de recuperación y la obtenida en las pruebas 1 y 2.

Para poder realizar la prueba de recuperación del estudiante deberá cumplir las tres condiciones siguientes:
. La nota de la asignatura es inferior a cinco.
. Como mínimo tiene un tres de las pruebas.
. Como mínimo tiene un tres de prácticas.

Recursos

Básicos

Bibliografías

- Booch, Grady. Análisi y Diseño Orientado a Objetos: con aplicaciones. 2da. Addison Wesley/Diaz de Santos, 1996. ISBN0-201-60122-2.
- Coad, Peter/ Yourdon Edward. Object Oriented Analysis. 2nd. Yourdon Press, 1991. ISBN0-13-629981-4
- Larman, Craig. UML and patterns: an introduction to analysis and object oriented design and the unified process. 2nd. Prentice Hall, 2003. ISBN9788420534381.
- Pressman, Roger S.. Software Engineering: a practical approach. 7. McGraw-Hill, 2010. ISBN 9786071503145.

Complementarios

Bibliografías

- Pressman, Roger S.. Software Engineering: a practical approach. 7. McGraw-Hill, 2010. ISBN 9786071503145.