

Diseño de Software

Código: 102759
Créditos ECTS: 6

Titulación	Tipo	Curso	Semestre
2502441 Ingeniería Informática	OB	3	1
2502441 Ingeniería Informática	OT	4	1

Contacto

Nombre: Joan Serrat Gual

Correo electrónico: Joan.Serrat@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

Equipo docente

David Fernández Mota

Idoia Ruiz López

Lluís Gesa Bote

Prerequisitos

No hay prerequisitos oficiales, pero es necesario que el estudiante domine la notación de diagramas de clase UML, los conceptos de orientación a objeto y su programación en algún lenguaje, preferiblemente Java.

Objetivos y contextualización

Una vez el propósito de un software se ha determinado mediante el análisis de requerimientos, los desarrolladores de software diseñan una solución. Esta va desde la arquitectura de sus componentes (clases, módulos ...) hasta la implementación en algún lenguaje de programación. Esta asignatura cubre principalmente dos aspectos claves del diseño: los patrones de diseño orientado a objeto y el diseño de la interfaz de usuario. Además de estos, trata otros temas como son el diseño por contrato de clases y métodos y el estilo de codificación.

El método de aprendizaje es basado en un proyecto (la práctica) o dicho de otro modo, 'aprender para hacer': todos los conceptos tratados en teoría son necesarios para diseñar e implementar una aplicación de software, a partir de un enunciado que hace el papel de documento de requerimientos. Se trata de hacer el diseño y la implementación (pero no el análisis de requerimientos, la prueba, el control de versiones o la garantía de la calidad, que pertenecen a otras asignaturas de la mención) de una aplicación para móvil en Android .

Competencias

- Ingeniería Informática
- Actitud personal.

- Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.
- 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.
- 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.
- 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.
- Trabajar en equipo.

Resultados de aprendizaje

1. Asumir y respetar el rol de los diversos miembros del equipo, así como los distintos niveles de dependencia del mismo.
2. Concebir sistemas software basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.
3. Conocer y aplicar los componentes de software específicos en problemas computacionales basados en comunicaciones.
4. Desarrollar la curiosidad y la creatividad.
5. Diseñar una interface de usuario partiendo de especificaciones del cliente.
6. Diseñar y documentar soluciones software.
7. Especificar las necesidades del cliente en un documento de especificación de software.
8. Identificar el paradigma que mejor se adapta al problema de diseño de un sistema informático software específico teniendo en cuenta los requerimientos de accesibilidad, ergonomía, usabilidad y seguridad.
9. Identificar, gestionar y resolver conflictos.
10. Mantener una actitud proactiva y dinámica respecto al desarrollo de la propia carrera profesional, el crecimiento personal y la formación continuada. Espíritu de superación.
11. Trabajar cooperativamente.

Contenido

1. Principios de orientación a objeto
2. Patrones de diseño: creacionales, estructurales, de comportamiento
3. Diseño por contrato
4. Estilo de codificación y comentarios. logging
5. Diseño de la interfaz de usuario
 - usabilidad
 - user research
 - proceso: prototipado, user testing
 - internacionalización
6. Programación de interfaces en Android

Metodología

Tal como hemos dicho a los objetivos, la asignatura sigue una metodología de aprendizaje basado en el desarrollo de un proyecto. Su implementación concreta en cuanto al uso de las horas de clase es la siguiente:

Clases de teoría. En ellas el profesor introduce los contenidos teóricos y da referencias útiles (libros, artículos, páginas web) para que el estudiante después pueda continuar su aprendizaje, así como desarrollar la aplicación software propuesta. Las transparencias utilizadas en estas clases pues no se deben tomar como fuente de estudio, sino que sirven para que el profesor pueda exponer de manera más ágil. En cambio, hay que estudiar los libros recomendados, y los artículos, capítulos de libros de acceso abierto y otros que ya indicaremos.

Clases de problemas. En ellas los estudiantes presentan las respuestas a una lista de cuestiones, que deberán resuelto previamente. Serán tanto problemas de diseño (diagramas de clase, código, prototipos de interfase de usuario etc.) como preguntas sobre artículos que habrá, por supuesto, haber leído y analizado, y que complementan los contenidos de las clases de teoría o obligan al su estudio.

Sesiones de prácticas. Los estudiantes deberán formado a principio de curso grupos pequeños (2 o 3 por grupo, si es posible). En las clases de prácticas los estudiantes vienen a 1) mostrarel trabajo al professor y resolver dudas, y 2) dado que la práctica estará dividida en hitos, las semanas que corresponden a un hito evalúa la parte correspondiente in situ, según un baremo que figurará en el enunciado de la práctica. Por tanto, las sesiones de prácticas *no* son para hacer la práctica.

Totos els materiales de la asignatura, junto con una planificación detallada se encuentran en el campus virtual. La mayoría son en inglés.

Actividades

Título	Horas	ECTS	Resultados de aprendizaje
Tipo: Dirigidas			
clases de problemas	12	0,48	2, 3, 6, 5
clases de prácticas	12	0,48	1, 2, 3, 6, 5, 9, 11
clases de teoria	26	1,04	2, 3, 6, 5, 8
Tipo: Autónomas			
estudio individual	29	1,16	3, 6, 5, 7, 8
preparación de prácticas	52	2,08	1, 2, 3, 6, 5, 8, 9, 11
resolución de problemas	12	0,48	2, 3, 6, 5, 7, 8

Evaluación

La forma de evaluación es continua. Esta consta de una parte de evaluación de grupo y otra individual. La nota de la primera (PR) se obtiene haciendo promedio de las notas de los hitos de prácticas (M1, M2, M3). La segunda (EX) del promedio de dos pruebas escritas durante el trimestre (EXP1, EXP2), o bien de un examen final de recuperación (EXR1, EXR2) de la parte o partes suspendidas exclusivamente. Los sufijos 1 y 2 se refieren a las 2 partes que tiene la asignatura, antes y después del primer examen EXP1.

El algoritmo para el cálculo de la nota final (FM) es el siguiente:

```
// EXP1, EXP2 exámenes parciales
// EXR1, EXR2 examen de recuperación
// M1, M2, M3 notas de las 3 metas de prácticas
// FM nota final
```

$EX = (\max (EXP1, EXR1) + \max (EXP2, EXR2)) / 2.$

```
if (M1> 0) and (M2> 0) and (M3> 0) {
PR = (M1 + M2 + M3) / 3.
} Else {
PR = 0.
}
```

```

if (EX >= 5.) and (PR >= 5.) {
  FM = (EX + PR) / 2.
} Else { // EX < 5. o PR < 5.
  FM = min (EX, PR)
}

```

Los hitos de prácticas son evaluaciones equivalentes al exámenes escritos. Si un estudiante no se presenta, la nota es cero.

La nota final será 'no evaluable' si no se ha entregado ningún examen ni ningún hito de prácticas.

No hay recuperación de prácticas. Dependiendo del año, podemos llegar a proponer ejercicios y problemas que sumen puntos a los exámenes EXP1, EXP2. Estos puntos se mantienen para los correspondientes exámenes de recuperación.

Algo que queremos asegurar mediante las diferentes evaluaciones es que cada miembro del grupo realmente haya contribuido a la práctica en la misma medida que los demás. Así pues, si como resultado de las preguntas que el profesor hace durante el hito o por las notas de pruebas escritas sobre la práctica, tenemos indicios de que esto no ocurre, entonces la nota de hito pasará de ser de grupo a individual: averiguaremos qué parte del hito actual ha hecho cada miembro y lo evaluaremos individualmente. Además, podemos llegar a dividir el grupo a partir de entonces.

Las fechas de evaluación continua y entrega de trabajos se publicarán en el campus virtual de la asignatura y pueden estar sujetos a cambios de programación por motivos de adaptación a posibles incidencias; siempre se informará a través de campus virtual, el mecanismo habitual de intercambio de información entre docentes y estudiantes.

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. Si el estudiante no se presenta en esta revisión, no se revisará posteriormente esta actividad.

No convalidamos prácticas, ejercicios ni exámenes de cursos anteriores. El único tratamiento diferenciado para los estudiantes repetidores es la posibilidad de entregar y evaluar todos los hitos de la práctica del curso actual (atención pues, no se trata simplemente de re-entregar la del curso pasado) en una misma fecha que ya se anunciaría.

Las notas MH las concederemos de manera discrecional a los estudiantes con nota final superior 9.0 y teniendo en cuenta todo tipo de trabajos hechos (exámenes, práctica y ejercicios).

Sin perjuicio de otras medidas disciplinarias que se estimen oportunas, se calificarán con un cero las irregularidades cometidas por el estudiante que puedan conducir a una variación de la calificación de un acto de evaluación. Por lo tanto, la copia, el plagio, el engaño, dejar copiar, etc. en cualquiera de las actividades de evaluación implicará suspender con un cero. 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, con nota cero.

Actividades de evaluación

Título	Peso	Horas	ECTS	Resultados de aprendizaje
evaluación de grupo	50%	4	0,16	2, 3, 4, 6, 5, 7, 8, 10
evaluación individual	50%	3	0,12	1, 2, 3, 4, 6, 5, 7, 8, 9, 10, 11

Bibliografía

Part 1

- Design patterns: elements of reusable object oriented software. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Addison Wesley, 1994. Versión en castellano de Pearson Educación, 2003.
- Design patterns explained simply. Alexander Shvets. <https://sourcemaking.com/design-patterns-ebook>.
- Head first design patterns. E. Freeman, E. Freeman, K. Sierra, B. Bates. O'Reilly, 2004.
- The elements of Java style. Al Vermeulen, Scott W. Ambler, Greg Bumgardner et al. Cambridge University Press, 2000.
- Clean code: a handbook of agile software craftsmanship. R.C. Martin. Prentice Hall, 2008. Edición en castellano Código limpio, d'Anaya Multimedia 2012.

Part 2

- The user experience team of one. A research and design survival guide. Leah Buley. Rosenfeld, 2013.
- Understanding your users. A practical guide to user research methods. Kathy Baxter, Catherine Courage, Kelly Caine. Morgan Kaufmann, segona edició 2015.
- Killer UX design. Jodie Moule. Sitepoint, 2012.
- Interviewing users. How to uncover compelling insights. Steve Portigal. Rosenfeld, 2013.
- Rocket surgery made easy: the do-it-yourself guide to finding and fixing usability problems. S. Krug. New Riders, 2010. Edición en castellano Haz facil lo imposible, d'Anaya Multimedia 2010.

Específicos para las prácticas:

- Thinking in Java, 3rd edition. B. Eckel. Versió pdf a <http://mindview.net/Books/TIJ/>. Versión en castellano de Pearson Educación, 2002.
- Beginning Android 4 application development. Wei-Meng Lee. Wiley, 2012.
- Head first Android development. Dawn Griffiths, David Griffiths. O'Reilly, 2015.