DEGREE CURRICULUM

SOFTWARE ENGINEERING

Coordination: SENDIN VELOSO, MONTSERRAT

Academic year 2022-23

SOFTWARE ENGINEERING 2022-23

Subject's general information

<table>
<tr><td>Subject name</td><td>SOFTWARE ENGINEERING</td></tr>
<tr><td>Code</td><td>102018</td></tr>
<tr><td>Semester</td><td>1st Q(SEMESTER) CONTINUED EVALUATION</td></tr>
<tr><td>Typology</td><td>
<table>
<tr><th>Degree</th><th>Course</th><th>Character</th><th>Modality</th></tr>
<tr><td>Bachelor's Degree in Computer Engineering</td><td>3</td><td>COMPULSORY</td><td>Attendance-based</td></tr>
<tr><td>Double bachelor's degree: Degree in Computer Engineering and Degree in Business Administration and Management</td><td>3</td><td>COMPULSORY</td><td>Attendance-based</td></tr>
<tr><td>Master's Degree in Informatics Engineering</td><td></td><td>COMPLEMENTARY TRAINING</td><td>Attendance-based</td></tr>
</table>
</td></tr>
<tr><td>Course number of credits (ECTS)</td><td>6</td></tr>
<tr><td>Type of activity, credits, and groups</td><td>
<table>
<tr><th>Activity type</th><td>PRALAB</td><td>TEORIA</td></tr>
<tr><th>Number of credits</th><td>3</td><td>3</td></tr>
<tr><th>Number of groups</th><td>2</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>Coordination</td><td>SENDIN VELOSO, MONTSERRAT</td></tr>
<tr><td>Department</td><td>COMPUTER SCIENCE AND INDUSTRIAL ENGINEERING</td></tr>
<tr><td>Teaching load distribution between lectures and independent student work</td><td>40% Presential (equivalent to 60h)
60% Autonomous work (equivalent to 90h)</td></tr>
<tr><td>Important information on data processing</td><td>Consult this link for more information.</td></tr>
<tr><td>Language</td><td>Preferably Catalan (Spanish if any student shows dificulties with Catalan).</td></tr>
<tr><td>Distribution of credits</td><td>Juan Manuel Gimeno Illa 4.5
Montserrat Sendin Veloso 4.5</td></tr>
</table>

SOFTWARE ENGINEERING 2022-23

Teaching staff	E-mail addresses	Credits taught by teacher	Office and hour of attention
SENDIN VELOSO, MONTSERRAT	montse.sendin@udl.cat	9	

Subject's extra information

Compulsory subject of 3rd year (1st quarter) that belongs to the common studies in the computer science branch.

Matter: Analysis and Design of Applications.

RECOMMENDATIONS: We assume the student knows the concepts about object-oriented programming and data structures teached in Programming II and Data Structures.

Learning objectives

- Knowing the conceptual basis and the different aspects of the discipline, among other the software lifecycle process model
- Apply the Use Case technique
- Specifying in a textual way the functional and non functional needs for a certain software system planned by means of a statement and/or other inputs from the user
- Developing the classes diagram for a certain software system following the Object Oriented Modeling principles
- Be familiar with a UML-based modeling tool
- Understanding the concept of code as a something that evolves over time
- Be able to program basic unit tests
- Understanding the object oriented design fundamental principles
- Recognizing the concept of responsibility as a fundamental one when planning an object oriented design

Competences

Cross-disciplinary competences

- **EPS-11:** Capacity to understand the needs of the user expressed in a no technical language

Specific competences

- **GII-CRI2:** Capacity to plan, conceive, deploy and direct projects, services and computer systems in all the fields, leading his set up and his continuous improvement and evaluation his economic and social impact
- **GII-CRI12:** Knowledge and application of the characteristics, functionalities and structure of the databases, that allow their suitable use, and the design and the analysis and implementation of applications based in them
- **GII-CRI13:** Knowledge and application of the necessary tools for the storage, processing and access to the Systems of information, including those based in web
- **GII-CRI16:** Knowledge and application of the principles, methodologies and life cycle of the software engineering
- **GII-CRI17:** Capacity to design and evaluate person-computer interfaces that guarantee the accessibility and usability of systems, services and computer applications.

Subject contents

SOFTWARE ENGINEERING 2022-23

Theme I - *Introductory aspects*

1.1. Initial questions about the Software Engineering

1.2. A little of history

1.3. Software development process

1.4. Software process models

1.5. Conclusions

Theme II - *Requirements Analisys*

2.1. Requirements specification

2.2. The *Use Cases technique*

2.3. A step more in the specification: the System Sequence Diagram

2.4. Conclusions

Theme III - *Domain Analisys*

3.1. *Analisys Classes Diagram*

3.2. A step more in the domain analysis: the *Contracts of the operations*

3.3. Conclusions

Theme IV - *Introduction to Design and Unit Testing*

4.1. The need for code design

4.2. The framework for unit testing JUnit 5

4.3. The version control system GIT

Tema V - *The SOLID principles*

5.1. Single responsibiliti principle

5.2. Open-closed principle

5.3. Liskov substitution principle

5.4. Interface segregation principle

5.5. Dependency inversion principle

Tema VI - *Responsibility based design*

6.1. The concept of responsibility

6.2. The GRASP patterns of responsibility assignement

Methodology

Big-size Groups: Masterly Classes (3 credits)

- Theorical part: Supported by snapshots and/or specific notes.
- Practical application part: Always working over examples. A **problems collection** is available. In class concrete problems are being solved. The solutions are being delivered along the semester.
- Putting in practice participatory and dynamic sessions **requires commitment** from the students.

Medium-size Groups: Laboratory Classes (3 credits)

- Guided classes and personalized monitoring through work teams in each medium-size group.
- **Cooperative Learning:** Work teams of **3 people** who collaborate with in the resolution of the course practice.
- UML Modeling tool usage: **ArgoUML** and/or **Visual Paradigm**.
- Control version tools with **GIT** and testing framework with **JUnit**.
- Progressive work regarding a certain **practical statement**, which will simulate the software project development as practical application of the subject contents.

Autonomous work (non presential):

- Practical work will be completed during **no presential** hours.
- **Highly recommended** to the student: solving the problems from the **collection**, in order to practise and get feedback from the teacher.

The **avaluation system** (detailed in el corresponding section) is composed of: **1)** writen tests (l2 partial exams); and **2)** practices (to develop in groups preferibly of two people).

Development plan

Week	Theory (GG)	Laboratory (LG)	Autonomous Work
1	Subject presentation T1: Introductory aspects	T1: Introductory aspects	Study
2	T1: Introductory aspects	T1: Introductory aspects	Study
3	T2: Requirements analysis Requirements specification	T2: Requirements analysis Requirements specification	Study and problems solving (Analysis problems collection)
4	T2: Requirements analysis. The Use Cases technique. Problems	UML Modeling usage Use Cases technique practical application	Study, problems solving (Analysis problems collection) and Analysis practice development
5	T2: Requirements analysis Use Case Specificaction Problems	Application of the Use Cases technique to the practice drafting T2: Requirements analysis. System Sequence Diagrams *Requirements analysis (1rst part)* *Delivery*	

6	T3: Domain analysis Object Oriented Modeling technique	SSD application to the practice drafting	Study, problems solving (Analysis problems collection) and Analysis practice development
7	T3: Domain analysis Object Oriented Modeling technique Problems	UML Modeling usage Practical application of the Object Oriented Modeling technique	Study, problems solving (Analysis problems collection) and practice defelopment *Requirements analysis (2nd part) Delivery*
8	T3: Domain analysis Object Oriented Modeling technique Problems	Construction of the Domain Model for the practice drafting	Study, problems solving (Analysis problems collection) and Domain analysis practice development
9	First midterm		Development of the Domain analysis practice
10	T3: Domain analysis Contracts of operations T4: Introduction to Design Test concept	Simple testing problems	Study, problems solving (Testing problems collection) and Domain analysis practice development
11	T4: JUnit Substitute objects	Testing with substitutions problems	Study and problems solving (Testing problems collection) *Domain Analysis and Contracts Delivery*
12	T5: Principis SOLID Intro, OCP & LSP	Testing problems. Advanced aspects of JUnit	Study, problems solving (Testing problems collection) and testing practice development
13	T5: Principis SOLID SRP, ISP & DIP	Git usage	Study, problems solving (Testing problems collection) and testing practice development
14	T6: Patrons GRASP Responsability concept	Git usage	Study, problems solving (Testing problems collection) and testing practice development
15	T6: GRASP Patrons Expert, Creator, Low Coupling	T6: GRASP Patrons High cohesion, Controller	Study and testing practice development
16	Second midterm		Testing practice development
17	Second midterm		*Testing Delivery*
18	Tutorization		
19	Recovery		

Evaluation

Activt.	Description	Weight	Mínimum Grade	In group	Presential	Mandatory	Recoverable
Part1	First midterm	30%	3,0	No	Yes	Yes	Sí
Part2	Second midterm	20%	No	No	Yes	No	Sí

Actv1	Requirements Analysis	20%	No	Yes	No	No	No
Actv2	Domain Model and Contracts	10%	No	Yes	No	No	No
Actv3	Unitary testing	20%	No	Yes	No	No	No

Final grade = 0,30 * Part1 + 0,20 * Part2 + 0,20 * Actv1 + 0,10* Actv2 + 0,20 * Actv3

- Subject is passed if **final grade** is greater or equal than **5,0** and the first midterm exam is above the minimum required.

Other considerations and criteria:

- Type of exams: concept fixation and problems solving.
- For all activities: programmed deliveries, unmovable dates.
- *Recovery exam:*
    1. **When the First midterm exam is below the minimum grade**, the student must do the recovey exam.
    2. **When the Final mark weighting is below 5**, even though reaching the minimum grade in the 1st midterm exam, the student must do the recovey exam.
    3. Furthermore, *it is a chance to improve the subject final mark, or to turn a compensable into a success (taking into account that the mark obtained in the recovery exam is the one that prevails).*
    - In these two last cases must be examined, at least, the midterm exam with the lower grade.
    - If even though, the mark is below the minimum grade required in the 1st midterm exam, **the final grade will be 4,5 at most.**

Bibliography

Basic bibliography

- Craig Larman, Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. Prentice-Hall, 2005 (3ª ed.)
- Boni García: Mastering Software Testing with Junit 5. Packt, 2017

Complementary bibliography

- Gerald Kotonya, Ian Sommerville: Requirements Engineering: Processes and Techniques. Wiley, 1998
- Robert Martin: Agile Software Development: Principles, Patterns, and Practices, Prentice-Hall, 2002
- Lasse Koskela, Effective Unit Testing. A guide for Java developers. Manning, 2013