

SUBJECT SYLLABUS

Degree				Academic year
144.2 BACHELOR'S DEGREE IN COMPUTER ENGINEERING				2012/13
Subject code and title				Duration
44220 Software Design				Semester 1
Type	Language	UD Credits	ECTS Credits	Group/Language
COMPULSORY	SPA-ENG	6	6	10 / English
Lecturer				
Cortazar Goicoechea, Rebeca				

DESCRIPTION

One of the key roles of graduates in Computing is the design and implementation of software solutions. If we analyse the life cycle of a software system, the second fundamental step is the design of the product to be built. In this course, students acquire the skills necessary for the design of distributed object-oriented software solutions, using UML as modeling notation and applying well-known design patterns, as well as heuristics and best practices. Therefore, this subject's contribution to the professional profile (from a competence perspective) is related to problem solving skills and system, component and application design, using a systemic approach (as well as creative and innovative), starting from existing requirements and taking into consideration different criteria for the evaluation of alternative solutions.

PREREQUISITES

Basic skills about thinking in objects, skills about programming using Java and basic concepts about RDBS.

LEARNING OUTCOMES IN TERMS OF GENERIC AND SPECIFIC COMPETENCES

GENERIC COMPETENCE GC6.2. Teamwork: Actively joining and participating in the attainment of shared objectives with other persons, departments and organisations. Level 2: Contributing to the consolidation and development of the team, fostering communication, balanced distribution of work, good team atmosphere and cohesion.

SPECIFIC COMPETENCE SC1. Design and evaluate alternative solutions to a software problem, applying patterns and design best practices.

SPECIFIC COMPETENCE SC2. Document software designs, correctly using suitable UML diagrams and notation.

SPECIFIC COMPETENCE SC3. Implement a software design, based on patterns, using distributed technologies.

CONTENTS

0. About Teamworking. Group Dynamics and Effective Teams. Roles in a Team. Conflict Management.
1. Architecture and UML Modeling. The Design stage in the Software Development Life Cycle. Concept of Architecture and Components. UML Component and Deployment diagrams notation.
2. Client-Server Applications. Characteristics. Distributed Objects: RMI. Persistence and Object-Relational Mappers: JDO.
3. Design Heuristics, Best Practices and UML Modeling. Design principles, Riel's Heuristics, GRASP Patterns, Best Practices and Refactorings. UML Sequence Diagrams.
4. Design Patterns. Enterprise Application Patterns, GoF Patterns, MS and J2EE Patterns. Antipatterns.

TEACHING-LEARNING STRATEGY

The teaching-learning strategy will be implemented by means of the following methods and techniques:

- Lectures: Driven by the professor, who will introduce the contents listed in the subject syllabus in a detailed and structured way in the classroom. Lecture materials, to be used during the lessons, will be previously available for students to read them in advance (slides, scripts, web links, etc.), organized by units.
- Case Study: Development of case studies, in order to assess alternatives and discuss design options. The most suitable alternative will be selected, depending on the context of the problem. During the development of those case studies, the notation of UML Sequence, Component and Deployment diagrams will be introduced.
- Personal Experimentation. Students will be provided with small problem statements so they can practice their modeling skills. On the other hand, lab sessions will be conducted in order to run basic RMI and JDO working examples, as well as get first-hand experience about design pattern implementation examples.
- Teamwork: Given a requirement specification, students will be requested to design a software solution based in patterns. They will have to use the UML notation to communicate and document their design and they will have to implement it using distributed technologies, RMI in particular, and use JDO for persistence.

According to the allocated 6 ECTS, the required time commitment for fulfilling this subject requirements is 150 hours, which will be distributed in agreement with the following working hours:

* Work in the classroom: 50 hours (lectures: 33 hours; labs: 17 hours)

* Work outside the classroom: 100 hours (teamwork: 68 hours, personal preparation: 26 hours, group support: 2 hours, assessment: 4 hours)

ASSESSMENT SYSTEM

- * Generic Competence: 10% of the final grade
- Peer-to-Peer Assessment (teamwork): 10%

- * Specific Competences: 90 % of the final grade.
- Knowledge test: 40%
- Teamwork: 50%

In order to pass the subject, at least 4 points out of 10 must be earned in each of the evaluation items: in the Knowledge test and in the teamwork coursework.

BIBLIOGRAPHY

- * Materials for the correct progress in the subject are on the e-learning platform.
- * The basic mean of communication will be the News and Forums in the platform; every event and news related to this subject will be posted on the platform.

References:

- * Gamma et al., Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional, ISBN-13: 978-0201633610.
- * A. Riel, Object-Oriented Design Heuristics, Addison-Wesley Professional, ISBN-13: 978-0201633856.
- * Craig Larman, Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Prentice Hall, ISBN: 978-0131489066.