

Academic Year: ( 2019 / 2020 )

Review date: 06-05-2020

Department assigned to the subject: Computer Science and Engineering Department

Coordinating teacher: GARCIA GUZMAN, JAVIER

Type: Compulsory ECTS Credits : 6.0

Year : 2 Semester : 2

**REQUIREMENTS (SUBJECTS THAT ARE ASSUMED TO BE KNOWN)**

Programming  
Software Engineering

**OBJECTIVES**

The goal of the course is to allow the student knowing in depth the software development process, with features about software testing and quality software issues.

To achieve this goal, the student must acquire the following PO: a, b, d, e, f, g, k

1. The transverse or generic competences that will be acquired throughout the course are

- 1.1 Abstraction capacity
- 1.2 Analysis and Synthesis Capacity (PO: b)
- 1.3 Capability to plan and organize (PO: b)
- 1.4 Ability to solve problems (PO: a, e, k)
- 1.5 Work in teams (PO: d)
- 1.6 Ability to apply knowledge in practice (PO: a, b, d, e, g, k)

2. The specific competences that will be developed throughout the course are:

- 2.1 Cognitive (Know) (PO: a, b, d, e, f, g, k) (CECRI8)
  - 2.1.a To know how to develop software components in a disciplined way including software development processes and lifecycle (CECRI16)
  - 2.1.b. Agile Software Development Culture
  - 2.1.c. Agile Software Development Techniques
  - 2.1.d. Test Driven Development
  - 2.1.e. Simple Design and Refactoring
  - 2.1.f. To know ethic features in software development process (CECRI11)
- 2.2 Instrumental (Know How) (PO :b, e, k)
  - 2.2.a. Develop a software component applying coding standards
  - 2.2.b. Develop a software component applying code collective ownership
  - 2.2.c. Develop a software component applying test driven development
  - 2.2.d. Develop a software component applying different testing techniques
  - 2.2.e. Develop a software component applying different automated testing
  - 2.2.f. Develop a software component applying continuous automated integration
  - 2.2.h. Develop a software component applying refactoring
  - 2.2.i. Develop a software component applying simple design patterns
- 2.3 Attitude (Be) (PO: e, k)
  - 2.3.a. Abilities to generate new ideas
  - 2.3.b. To develop a good work using quality measures
  - 2.3.c. Problem solving orientation
  - 2.3.d. To investigate and find solutions to new problems
  - 2.3.e. To develop a good work using quality measures

**DESCRIPTION OF CONTENTS: PROGRAMME**

- 1.- Ethic and Legal Issues in the Software Engineering Profession
  - 1.1.- The software engineering profession.
  - 1.2.- The software engineers' code of ethics.
- 2.- Agile Software Development Techniques
  - 2.1.- Pair Programming

- 2.2.- Coding Standards
- 2.3.- Code Collective Ownership
- 3.- Test Driven Development
  - 3.1.- Principles of Test Driven Development
  - 3.2.- Functional Testing Techniques
  - 3.3.- Estructural Testing Techniques
  - 3.4.- Unit Testing Automation
  - 3.5.- Automated Continuous Integration
- 4.- Refactoring and Simple Design
  - 4.1.- Refactoring
  - 4.2.- Principles of Software Design
  - 4.3.- Design Patterns for Responsibilities Assignment

## LEARNING ACTIVITIES AND METHODOLOGY

Lectures: 1,5 ECTS, to get the specific cognitive and instrumental competences of the subject. (PO: a, b, e, f, g, k)

Exercise Classes: 1,5 ECTS, to get the specific instrumental and generic competences, as well as the attitude competences of the subject. A practical example related to the development of a software component (including estimation, specification, design, software reviews and testing activities) will be carried out, considering ethic and legal features. (PO: a, b, d, e, f, g, k)

Academic Work with professor assistance: 1 ECTS, Presentation and discussion of a specific software component development case, analyzing all the aspects considered in the theoretical part of the subject. (PO: a, b, d, e, f, g, k)

Academic Work without professor assistance: 1 ECTS, Individual work for the development of a specific software component development case, analyzing all the aspects considered in the theoretical part of the subject. (PO: a, b, e, f, k)

Exam: 1 ECTS (PO: a, b, d, e, f, g, k)

## ASSESSMENT SYSTEM

Guided Exercises 30% (PO: a, b, d, e, f, g, k)

Theory Tests 30% (PO: a, b, d, e, f, g, k)

Final Practice (Exam) 40% (PO: a, b, d, e, f, g, k)

It is mandatory to pass, separately, each one of the parts (theory, guided exercises and final practice) to pass the whole subject

<b>% end-of-term-examination:</b>	40
<b>% of continuous assessment (assignments, laboratory, practicals...):</b>	60

## BASIC BIBLIOGRAPHY

- Beck, Ken Extreme Programming Explained, Addison-Wesley, 2000
- Beck, Ken, et al. Test-Driven Development By Example, Three Rivers Institute, 2002
- Craig S. Larman Applying UML and Patterns, Pearson Education , 3er Edition, 2012
- Fowler, Martin et al. Refactoring: Improving the Design of Existing Code, Addison-Wesley, 1999
- Lee Copeland A Practitioner's Guide to Software Test Design. , Artech House Publishers .

## ADDITIONAL BIBLIOGRAPHY

- Paul C Jorgensen Software Testing: a craftsman's approach, CRC Press.
- Roger S. Pressman Software Engineering. A practical Approach, McGraw Hill, 7th Edition, 2009
- Spyros Xanthakis, Michel Maurice, Antonio de Amescua, Olivier Houri, Luc Griffet Test & Controle des Logiciels: Methodes, Techniques & Outils, EC2.