Software Development

Academic Year: (2021 / 2022) **Review date: 30-06-2021**

Department assigned to the subject: Computer Science and Engineering Department

Coordinating teacher: LOPEZ CUADRADO, JOSE LUIS

Type: Compulsory ECTS Credits : 6.0

Year : 3 Semester : 2

REQUIREMENTS (SUBJECTS THAT ARE ASSUMED TO BE KNOWN)

Programming (1º,1)
Data structures & Algorithms (1º,2)
Software Engineering (2º,1)
Automata and formal language theory (2º,1)

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.

The learning results will be:

R1 (Knowledge and understanding): Have basic knowledge and understanding of the scientific and technological foundations of Computer Engineering, as well as specific knowledge of computer science, computer engineering and information systems).
R2 (Engineering Analysis): Be able to identify Computer Engineering problems, recognize their specifications, establish different resolution methods and select the most appropriate for their solution, taking into account applicable social, human health, environmental, and commercial limitations for each case).
R3 (Engineering Design): Be able to make engineering designs according to the student's level of knowledge and understanding that meet the required specifications, collaborating with other engineers and graduates. The design covers devices, processes, methods and objects, and broader specifications than strictly technical, which includes social awareness, health and safety, and environmental and commercial aspects.

Basic and General Competencies
CB2 - That students know how to apply their knowledge to their work or vocation in a professional manner and possess the competencies that are usually demonstrated by the elaboration and defense of arguments and the resolution of problems within their area of study.
CG5 - Use computer-based, general purpose, collaborative, and work optimization tools for effective project planning and implementation.
effective implementation of projects.
CG9 - Efficiently use ICT means to write technical reports and project and work report, as well as quality presentations.
CGO3 - Ability to design, develop, evaluate and ensure the accessibility, ergonomics, usability and security of systems, services and applications, as well as of the information they manage.
CGO5 - Ability to conceive, develop and maintain computer systems, services and applications using the methods of software engineering as a tool for quality assurance, according to the knowledge acquired.
CGO7 - Ability to know, understand and apply the necessary legislation during the development of the profession of Computer Engineering, and to handle specifications, specifications, services and applications.
Technician in Computer Science and to handle specifications, regulations and standards of mandatory compliance.

Specific competences
CECRI1 - Ability to design, develop, select and evaluate computer applications and systems, ensuring their reliability, security and quality, according to ethical principles,
security and quality, in accordance with ethical principles and current legislation and regulations.
CECRI8 - Ability to analyze, design, build and maintain applications in a robust, secure and efficient

way, choosing the right paradigm and programming languages.
paradigm and the most appropriate programming languages.
CECRI16 - Knowledge and application of software engineering principles, methodologies and life cycles.

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.- Software development process
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.

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.

Practical Work with and without professor assistance: 2 ECTS, work on specific software component development case, analyzing all the aspects considered in the theoretical part of the subject.

Tutorials: Individualised assistance (individual tutorials) or group assistance (group tutorials) to students by the lecturer.

Exam (Final Practice): 1 ECTS

ASSESSMENT SYSTEM

The exercises and exams, in addition to serving as a training activity, have the double objective of being measured by the evaluation system. The evaluation system includes the evaluation of the directed academic and practical activities according to the following weighting.

Guided Exercises (Continuous assessment) 30%
Theory Tests/Exercises (Continuous assessment) 30%
Final Practice (Exam) 40%

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

% end-of-term-examination:	40
% of continuous assessment (assigments, laboratory, practicals…):	60

BASIC BIBLIOGRAPHY

- Beck, Ken, et al.. Test-Driven Development By Example. , Three Rivers Institute., 2002
- Beck, Ken. Extreme Programming Explained., Addison-Wesley. , 2000
- 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, 2003

ADDITIONAL BIBLIOGRAPHY

- Paul C. Jorgensen Software Testing: a craftsman's approach. , CRC.

- Roger S. Pressman. Software Engineering. A practical approach., McGraw Hill. 7ª Edición., 2009

- Spyros Xanthakis, Michel Maurice, Antonio de Amescua, Olivier Houri, Luc Griffet. Test and contro¿le des logiciels : me¿thodes, techniques and outils, EC2..