SUBJECT TEACHING GUIDE

G668 - Methods of Development

Degree in Computer Systems Engineering
First Degree in Computer Systems Engineering

Academic year 2024-2025

1. IDENTIFYING DATA					
Degree	Degree in Computer Systems Engineering First Degree in Computer Systems Engineering			Type and Year	Optional. Year 4 Optional. Year 4
Faculty	Faculty of Sciences				
Discipline	Subject Area: Software Engineering Mention in Software Engineering				
Course unit title and code	G668 - Methods of Development				
Number of ECTS credits allocated	6	Term		Semester based (1)	
Web	http://moodle.unican.es				
Language of instruction	Spanish	English Friendly	No	Mode of delivery	Face-to-face

Department	DPTO. INGENIERÍA INFORMÁTICA Y ELECTRÓNICA
Name of lecturer	PABLO SANCHEZ BARREIRO
E-mail	p.sanchez@unican.es
Office	Facultad de Ciencias. Planta: + 1. DESPACHO PROFESOR (1069)
Other lecturers	JULIO LUIS MEDINA PASAJE JUAN MARIA RIVAS CONCEPCION

3.1 LEARNING OUTCOMES
- To be able to develop software products following a methodology
- To be able to use one of the main software development paradigms: model-driven development, component-based, aspect-oriented, event-based, etc.
- To be able to use the main techniques and regulations to elaborate a plan for a development or maintenance project.
- To understands how a software process works and to be able to model and specify them.
- To use techniques and tools for configuration management.
- To use methodologies and techniques to develop modernization projects or services.
- To be able to use the main methods, techniques and tools for software testing, verification and validation.
- To be able to use methods and techniques for software project management and monitoring.
- To be able to estimate size,effort and cost of a software project.
- To be able to perform a risk analysis and elaborate a risk mitigation and control plan for a software development project or for a software system in operation.

4. OBJECTIVES
To be able to manage software system configurations using a software configuration manager like Git.
To understand how a software configuration management process works.
To understand the differences between heavyweight and agile methodologies.
To understand the foundations of agile methodologies.
To understand the main techniques of agile methodologies.
To understand the similarities and differences between the main agile methodologies.
To be able to develop a medium-size software system using an agile methodology like Scrum.
To be able to model software development processes using a software process modelling language like SPEM.
To understand how a software maintenance process work.
To understand how the Métrica v3 software development process works. NOTE: Métricav3 is the software development methodology used by most of the Spanish Institutions and Public Services.
To be able to use professional branching schemes such as GitFlow.
To understand how continuous integration works.
To undestand what DevOps is.

6. SUBJECT PROGRAM

CONTENTS	
1	Unit 1. Software Configuration Management Introduction. Terminology. Software Configuration Management Processes. Continuous Integration and Continous Delivery. Version Control with Git. Branching Schemes: GitFlow. DevOps.
2	Unit 2. Agile Methodologies. Heavyweight and Agile Methodologies. Agile Manifesto. Lean Principles. Foundational Techniques of Agile Methodologies: User Stories, Test-Driven Development, Planning Poker, Pair-Programming. Scrum. Additional Agile Methodologies: XP, Kanban.
3	Unit 3. Software Process Modelling Definition of Software Process. Software Process Modelling with SPEM.
4	Unit 4. Regularized Software Methdologies Métrica v3. Software Maintenance Processes in Métrica v3.

7. ASSESSMENT METHODS AND CRITERIA

Description	Type	Final Eval.	Reassessn	%
Lab Assignments	Laboratory evaluation	No	Yes	30,00
Development of a Software Project using Scrum.	Work	No	Yes	70,00
TOTAL				100,00

Observations

When the mark of some qualifiable element is lower than the required minimum, but the weighted average of all qualifiable elements is greater or equal than five, the final mark will be 4.9, i.e., the course will be failed.

Instructors might perform extra checks to verify the authority of the assignments delivered by the students. Plagiarism is not allowed and it will imply that the student will fail the course. In addition, the plagiarism will be notified to the Faculty Council so that disciplinary actions can be adopted.

When the mark for the project to be developed is lower than 5.00, the student, the student will have to pass the final test to pass the subject. In these cases, the mark of this test will be the final mark.

Observations for part-time students

The main goal of this subject is that students learn to develop software projects in teams and following a well-defined software methodology. Thus, interacting with course mates become mandatory, being these interactions an evaluable item. Therefore, there is no option for developing this course project individually, so part-time students must participate in work teams, which can
be comprised of full-time and part-time students.

A minimum attendance is required since each student needs to interact with her team . The work in teams is limited to weeks 6th to 11th. Thus, it is recommended that all students can assist to the classroom during these weeks. This minimum attendance will be established at the beginning of the semester, so that each student can make the arrangements she needs as soon as possible. Under some circunstances, virtual attendance might be allowed to part-time students.

8. BIBLIOGRAPHY AND TEACHING MATERIALS
BASIC
Mike Cohn. "User Stories Applied:For Agile Software Development". Addison-Wesley. Marzo 2010.