

BASIC DETAILS:

Subject:	CALIDAD DEL SOFTWARE		
Id.:	31844		
Programme:	DOBLE GRADO EN INGENIERÍA INFORMÁTICA Y DISEÑO Y DESARROLLO DE VIDEOJUEGOS		
Module:	INGENIERIA DEL SOFTWARE		
Subject type:	OBLIGATORIA		
Year:	3	Teaching period:	Segundo Cuatrimestre
Credits:	3	Total hours:	75
Classroom activities:	35	Individual study:	40
Main teaching language:	Inglés	Secondary teaching language:	Inglés
Lecturer:		Email:	

PRESENTATION:

Day by day the complexity and amount of problems to be solved via Software is increasing. Unfortunately, the software tools, methodologies and techniques are not evolving at the same pace. Due to this, guaranteeing the Quality of the Software has become more complicated. Hence, In a global world, where competition is so strong, having products of better quality than the competitors is essential.

However, the first challenging question is understanding what is Software Quality. We will work to identify different ways to assess Software Quality, especially paying attention to the customers point of view, as they are the ones that make a Software Product meaningful.

We will also review different metrics that can be useful in order to assess the quality of a product as well as the quality of the process a product was built with.

We will present how the dynamic nature of Software requires a suitable set of tools that allow the Software to be managed, allowing many people to work together in the same software and in different versions.

Last but not least, we will review the activities that allow to assure the quality of a product (e.g. check that the product meets some expectations) with special emphasis in Testing.

PROFESSIONAL COMPETENCES ACQUIRED IN THE SUBJECT:

General programme competences	G01	Leadership capacity to be able to influence a group so they achieve some specific objectives collectively and efficiently.
	G02	Innovative capacity to propose and find new and efficient ways to undertake any task and/ or function within the professional environment - highly motivated by quality.
	G03	Capacity to work in multidisciplinary teams to achieve common objectives, placing group interests before personal ones.
	G04	Capacity to always commit to working responsibly - creating a strong sense of duty and fulfilment of obligations.
	G05	Capacity to adapt to different environments while being positive and optimistic, orienting your behaviour towards the achievement of goals.
	G06	Capacity to analyse and find a solution to complex problems or unforeseen situations which may arise while working in any type of socio-economic organisation.
	G07	Capacity to work flexibly and with versatility to adapt to the needs and requirements of the work situation.
	G08	Ability to communicate effectively about different matters in a variety of professional situations and with the different media available.
	G09	Capacity to make decisions impartially and rationally.
	G10	Critical and analytical capacity when assessing information, data and courses of action.
	G11	Ability to get on in a multicultural or international environment, interacting with people of different nationalities, languages and cultures.
	G12	Capacity to undertake professional activities with integrity, respecting social, organisational and ethical norms.

	G13	Capacity to use individual learning strategies aimed at continuous improvement in professional life and to begin further studies independently.
Specific programme competences	E04	Capacity to maintain an open mind to innovation and creativity within the framework of the engineering profession.
	E05	Capacity to assess the economic and business features of engineering activities.
	E06	Capacity to apply quality assurance processes to processes and products.
	E07	Capacity to work effectively in project teams, where appropriate assuming executive responsibilities, and consider the human, technological and financial sides.
	E08	Capacity to communicate productively with clients, users and colleagues both orally and in writing, so as to pass on ideas, solve conflicts and achieve agreements.
	E09	Capacity to maintain professional competences through independent learning and continuous improvement.
	E12	Capacity to manage complexity through abstraction, modelling, 'best practices', patterns, standards and the use of the appropriate tools.
	E15	Capacity to understand and go along with the strategic objectives of the company where you are pursuing your professional career.
	E16	Capacity to understand an application domain so as to be able to develop suitable IT applications.
	E17	Capacity to identify and analyse user needs with the intention of designing effective, usable IT solutions which can be incorporated into the user's operating environment.
	E18	Capacity to identify and define the requirements to be satisfied by IT systems to cover the stated needs of organisations or individuals.
	E19	Capacity to design and define the architecture of IT systems (software, hardware and communications) under the requirements agreed upon by the parties involved.
	E20	Capacity to undertake the detailed design of the components of a project (procedures, user interface, equipment characteristics, communications system parameters, etc.).
	E21	Capacity to perform tests that verify the validity of the project (functional, data integrity, performance of the computer applications, equipment, communications, etc.).
	E22	Capacity to undertake implementation tasks which require a high degree of technical awareness in different spheres (programming, configuration of hardware and communications equipment, etc.).
	E24	Capacity to draw up and develop effective project plans for systems based on information and communication technologies.
	E25	Capacity to analyse viability, design development plans, estimate resources, run and oversee the execution of software-intensive engineering projects.
	E26	Capacity to define and manage quality policies for IT and communications systems, applying quantitative principles based on metrics and statistics.
	E27	Capacity to write and maintain descriptive documentation of the origin, production and operability of IT systems.

PRE-REQUISITES:

"Software Engineering"

SUBJECT PROGRAMME:

Observations:

- Understand what is Quality and Software Quality and how quality is part of the Software Development (and Delivery) process.
- Learn how to measure some attributes related to Quality.
- Learn about which activities can be carried out to increase Software Quality (preventing defects, removing them or minimizing the impact on end-users)
- Learn which tools are available and should be used to deliver high quality software.

Subject contents:

1 - Introduction to Software Quality
1.1 - What is Software Quality
1.2 - Software Quality Activities
1.3 - Software Quality Engineering
2 - Software Quality Metrics
2.1 - Product Metrics
2.2 - Process Metrics
2.3 - Metrics in Object Oriented Programming
3 - Software Configuration Management
3.1 - Basic Principles
3.2 - Patterns
3.3 - Continuous Integration
3.4 - Other Tools
3.5 - SCM in practice: Git
4 - Testing
4.1 - Basic Concepts
4.2 - Key Activities
4.3 - Test Coverage
4.4 - Input Domain Partition and Boundary Testing
4.5 - Flow Control, dependencies and interactions
4.6 - Patterns
4.7 - Frameworks
4.8 - Test Driven Development
5 - QA Activities beyond Testing
5.1 - Defect Prevention and Process Improvement
5.2 - Code Inspection and Formal Verification
5.3 - Assertion Driven Development and Design by Contract
5.4 - Error Tolerance and Failure Contention

Subject planning could be modified due unforeseen circumstances (group performance, availability of resources, changes to academic calendar etc.) and should not, therefore, be considered to be definitive.

TEACHING AND LEARNING METHODOLOGIES AND ACTIVITIES:

Teaching and learning methodologies and activities applied:

- Master classes: Teacher will use these to solve doubts about lessons that should have been studied before, assignments as well as presenting new subjects for the following weeks.
- Tutoring Sessions: These sessions will be used for having more interactive sessions with the students, especially about practical cases, assignments or exercises.
- Practical Cases: These sessions are intended to learn a specific tool or a best practice, for instance, learning how to use git.
- Problem Resolution: These sessions will be used for resolving practical exercises.

It is recommended to students:

- to be up-to-date with the subject, following the topics that are presented by the teacher in every session.
- To participate actively in the classes, asking questions or raising any issue in a timely manner.
- To resolve optional exercises and assignments as suggested by the teacher.

The attendance rules and the corresponding penalties will be strictly enforced as described in section 4.5 of the academic guide for degree programs 2018-2019 (pages 29, 30 and 31).

Students won't be allowed to enter the classroom after the beginning of the class or leave before it ends without a clear and fair justification (can be considered either of the two alleged lack of assistance)

Student work load:

Teaching mode	Teaching methods	Estimated hours
Classroom activities	Master classes	20
	Practical exercises	5
	Practical work, exercises, problem-solving etc.	5
	Other practical activities	2
	Assessment activities	3
Individual study	Individual study	16
	Individual coursework preparation	14
	Group coursework preparation	5
	Research work	5
Total hours:		75

ASSESSMENT SCHEME:

Calculation of final mark:

Written tests:	10	%
Individual coursework:	40	%
Group coursework:	10	%
Final exam:	30	%
Participation:	10	%
TOTAL	100	%

*Las observaciones específicas sobre el sistema de evaluación serán comunicadas por escrito a los alumnos al inicio de la materia.

BIBLIOGRAPHY AND DOCUMENTATION:

Basic bibliography:

Fowler, Martin. "Refactoring", Pearson Addison Wesley, 2018
Horch, John W. "Practical Guide to Software Quality Management", Artech House, 2003
Humble, Jez & Farley, David. Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation, Addison-Wesley, 2010
Martin, Robert C. "Clean Code", Prentice Hall, 2008
Nygard, Michael, "Release It!: Design and Deploy Production-Ready Software", O'Reilly, 2018
Patton, Ron. "Software Testing", 2nd ed., Sams Publishing, 2005

Recommended bibliography:

Burnstein, Ilene. "Practical Software Testing", Springer-Verlag, 2003.
Evans, Isabel (2004). "Achieving Software Quality through Teamwork", Artech House.
Kan, Stephen (2002). "Metrics and Models in Software Quality Engineering", Addison Wesley
Lewis, William. "Software Testing and Continuous Quality Improvement" 2nd ed., Auerbach Publications, 2005.
Ravindranath, Pandian. "Software Metrics. A Guide to Planning, Analysis and Application", Auerbach Publications, 2004
Stamelos, Ioannis G. "Agile Software Development Quality Assurance", Idea Group Inc, 2007.
Tian, Jeff. "Software Quality Engineering. Testing, Quality Assurance and Quantifiable Improvement", John Wiley, 2005.

Recommended websites:

Algunos blogs relacionados con la calidad del software	http://jmbeas.iexpertos.com/ http://adam.goucher.ca/ http://www.codinghorror.com/blog/
Este portal surge en el año 2004 a raíz del éxito del grupo Yahoo del mismo nombre, con la intención de ser un sitio donde las persona interesadas en la mejora de la calidad del software pudieran encontrar noticias, artículos e información general sobre	http://blog.utest.com/ https://testing.googleblog.com/ https://www.satisfice.com/blog/
La Asociación Española para la Calidad (AEC) es una entidad privada sin ánimo de lucro, fundada en 1961, cuya finalidad es fomentar y apoyar la competitividad de las empresas y organizaciones españolas, promoviendo la cultura de calidad y desarrollo sostenible	https://www.stickyminds.com/
Software Fault Tolerance: A tutorial (NASA).	http://calidaddelsoftware.com/
	http://www.aec.es/
	http://www.tpub.com/content/nasa2000/NASA-2000-tm210616/NASA-2000-tm2106160001.htm

* Guía Docente sujeta a modificaciones