Universitat Autònoma de Barcelona

2022/2023

Software Quality and Testing

Code: 102758
ECTS Credits: 6

Degree	Type	Year	Semester
2502441 Computer Engineering	OB	3	1
2502441 Computer Engineering	OT	4	1

Contact

Name: Xavier Otazu Porter
Email: xavier.otazu@uab.cat

Use of Languages

Principal working language: catalan (cat)
Some groups entirely in English: No
Some groups entirely in Catalan: Yes
Some groups entirely in Spanish: No

Teachers

Antonio Ballester Valiente

Prerequisites

It is not obligatory, but it is recommended to study in parallel the subject "Software Requirements" and "Software Design".

Objectives and Contextualisation

In this subject we introduce the basic concepts to plan software tests and the techniques to implement them, as well as estimates of their quality.

Competences

Computer Engineering

- Ability to develop, maintain and evaluate software services and systems that meet all user requirements and behave reliably and efficiently, are affordable to develop and maintain and meet quality standards, applying theories, principles, methods and practices of the Software engineering.
- Acquire personal work habits.
- Have the capacity to conceive, develop and maintain computer systems, services and applications employing the methods of software engineering as an instrument to ensure quality.

Learning Outcomes

1. Apply formal methods of software design review.
2. Apply formal techniques to the verification of compliance with the requirements of software applications.
3. Critically evaluate the work done.

4. Evaluate software systems ensuring that they meet all user requirements and behave reliably and efficiently, are affordable to develop and maintain and meet quality standards, applying the theories, principles, methods and practices of Software Engineering.
5. Plan and document the software design in practical cases.

Content

The temporal order of the following contents throughout the course does not necessarily follow this order.

Introduction to software testing

Objective of the test. Types of tests: white box, black box, validation, regression, loading, etc. Test levels: unity, integration, system, acceptance.

Unit tests and integration tests

Objective and methodology. Why test for unity and integration? Unit Testing and Test Driven Development. "Mock" objects.

Black box tests

Equivalence classes. Limit value analysis. Decision tables. Pairwise testing. Transition of states. Design of test cases from use cases. Example ATM.
Test based on use cases.

White box tests
Flow control (cyclomatic complexity). Define strings. Coverage test versus flow control test.

Exploratory test
Simple and fast tests. Practical interest and effectiveness.

Test automation

Creation of "automatic" tests. Web and recording tools (Selenium).

System testing or End-to-End testing

Objective and methodology. Why do system tests? Behavior Driven Development (BDD). Review of Test Reports.

Test Frameworks

Definition and examples. Implementation of a test framework and system tests using Cucumber (BDD).

Test planning and documentation

Risk analysis. Mastertestplan. Examples of test plan and documentation.

Quality control, Quality Assurance and Quality Engineering

Formal reviews, metrics, development standards.

Bug definition and bug reports. Bug Tracking Tools: Bugzilla and Jira. How to Integrate a Quality Engineer into a Development Team. Formal technical reviews.

Ecosystem of a Quality Engineer

Definition of the DevOps cycle. Continuous Integration, Continuous Testing, Continuous Delivery, Continuous Feedback. Best-known tools (Jenkins, GitHub / GitLab and change request reviews, Grafana).

Methodology

This subject has a balance between the theoretical part and the practical part. It is not a totally ABP subject matter since the theoretical concepts are important, and the practical part will be carried out with tutorial sessions, delivery sessions and evaluation of the planetary problems and autonomous work on the part of the student.

Theory classes: These are master classes where the theoretical concepts of the subject are introduced. Although theory classes, a good part of the session (approximately half) will be used to solve in a group simple problems raised by the teacher.

Seminars: These are classes where practical cases will be considered that will require the application of the concepts explained in the theory classes. At the end of the class of seminars there will be a problem that the student will have to solve autonomously and that he will deliver voluntarily at the beginning of the next session of seminars.

Laboratories: These are sessions in which the teacher will conduct tutorials with the students in order to guide their work while developing the solution to practice a real problem of a certain complexity. Before these sessions, the student will have to perform a series of tasks autonomously that will be shown at the beginning of the laboratory session.

The transversal competence, T02.08 - Critically evaluate the work carried out, will be evaluated automatically in the practices, since this is a necessary capacity to be able to perform a practice autonomously.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Theory sessions	22	0.88	2
Type: Supervised			
Laboratories	12	0.48	2, 1, 3, 4, 5
Seminars	12	0.48	2, 3, 4, 5
Type: Autonomous			
Evaluation tests study	60	2.4	2
Laboratories preparation	24	0.96	2, 4, 5
Seminars preparation	14	0.56	2, 4

Assessment

The final grade of the subject is given by the formula: Note = 0.55 * NTeo + 0.45 * NLab,

NTeo is the note of Theory, and NLab the note of laboratories. Each of the NTeo and NLab notes must be higher than 5, otherwise the final grade of the subject will be the minimum value of these two. The Note value will only be calculated if NTeo and NLab are both higher than 5. In the event that it does not, the Note value will be the minimum value of NTeo and NLab.

NTeo note is

NTeo = 0.5 * NTeo1 + 0.5 * NTeo2

where NTeo1 and NTeo2 are two evaluation activities that will be carried out throughout the course in hours of theory sessions. The value of NTeo will only be calculated if NTeo1 and NTeo2 are both higher than 4. In the event that it does not, the value of NTeo is considered to be the minimum value of NTeo1 and NTeo2. If NTeo is not more than 5 after performing both NTeo1 and NTeo2 tests, the student can re-evaluate NTeo1 and / or NTeo2 during the final assessment tests. In the final assessment tests, each of the NTeo1 and NTeo2 notes will be saturated at a value of 6

At the same time of publication of each evaluation, a period will be given to review the note. Scores of qualification review will be scrupulously respected, that is, no qualification will be reviewed outside this note review period.

The NLab will be a weighted average of the different NLab (i) laboratory sessions.

NLab = w1*NEntr1 + w2*NEntr2

where NEntr (i) is the individual note of each of the two practice deliveries (expected in the laboratory sessions 3 and 6 of the course, respectively), and w1 and w2 are the weights (to be announced on the first course day). The calculation of these notes is specified in the statement of each practice (to be published in the Moodle classrooms)

There is no second call for the NLab note or any ofthe NLab (i) notes. The value of NLab will only be calculated if eachandevery one of NLab(i) is equal to or greater than 0.1. Otherwise NLab will be the minimum value among all NLab (i) values.

The order of assignment to the groups of practices will be done according to the note of the academic file.

The NLab note will be validated for those students who have studied the subject at a maximum of one previous year. No other note will be validated.

A student will obtain a "Non-Valuable" as the final grade of the subject only in the event that he does not obtain a qualification in any of the variables that appear in the previous mathematical formulas. If a student presents some work to evaluate or perform some of the assessment activities that are contemplated in the previous formulas, they will no longer be able to obtain a "Non-Valuable".

To obtain a Honor Matriculation, it is necessary, as a mandatory requirement, to obtain a final qualification of Excellent.

The dates of continuous assessment and work delivery will be published on the virtual campus (CV) or Moodle (MO) classrooms and may be subject to changes of programming for reasons of adaptation to possible incidents. The CV / MO will always be informed about these changes since it is understood that CV / MO is the usual mechanism for exchanging information between teacher and students.

Notwithstanding other disciplinary measures deemed appropriate, and in accordance with the current academic regulations, irregularities committed by a student that can lead to a variation of the qualification will be classified by zero (0). For example, plagiarizing, copying, copying, ..., an evaluation activity, will imply suspending this evaluation activity with zero (0). Assessment activities qualified in this way and by this procedure will not be recoverable. If you need to pass any of these assessment activities to pass the subject, this subject will be suspended directlywith a qualification of zero (0), without the opportunity to recoveritin the same course.

These irregularities include, among others:

the total or partial copy of a practice, report, or any other evaluation activity;

let copy;
present a group work not done entirely by the members of the group (applied to all members, not only to those who have not worked);
present as own materials prepared by a third party, even if they are translations or adaptations, and generally works with non-original and exclusive elements of the student;
Have communication devices (such as mobile phones, smart watches, camera pens, etc.) accessible during theoretical-practical assessment tests (individual exams);
Talk with classmates during the theoretical-practical assessment tests (exams);
Copy or attempt to copy from other students during the theoretical-practical assessment tests (exams);
Use or attempt to use written material related to the subject during the theoretical-practical evaluation tests (exams), when these have not been explicitly allowed.

Even a student who has committed irregularities in an evaluation act will not be validated

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Individual continuous assessments Theory 1	0.225	3	0.12	2
Individual continuous assessments Theory 1	0.225	3	0.12	2
Laboratories evaluation	0.55	0	0	2, 1, 3, 4, 5

Bibliography

[1] A Software Testing Primer: an Introduction to Software Testing. Nick Jenkins. http://www.nickjenkins.netLectura obligada!
[1,2,3,5,6] Manage Software Testing. Peter Farrell-Vinay. Auerbach Publications, 2008.
Esta bien la parte de documentación y planning, però la de tècnicas de prova no tanto.
[1,3,4] A practitioner's guide to software testing. Lee Copeland, 2004.
Libro muiy sencillo y breve (quizás demasiado), enfoque práctico.
[4] How to break software: a practical guideto testing. James A. Whittaker. Addison Wesley, 2003.
EL libro sobre prueba exploratoria.
[5] Pragmatic Unit Testingin Java with Junit. Andy Hunt, Dave Thomas.The Pragmatic Programmer, 2003.
[6] IEEE Standard for Software and System Test Documentation IEEE Std82-2008. IEEE Computer Society, 2008.
[6] Systematic Software Testing by. RickD. Craig, Stefan P.Jaskiel. ArtechHouse, 2002.
El mejor libro que he encontrado sobre la planificación de pruebas. Incluye una explicación del estándard IEEE de documentación.
[7] Llibre EUIS, veure a dalt.
[8] Why the Vasa sank: 10 problems and some antidotes for software projects. Richard E. Fairley, Mary Jane Willshire. IEEE Software, març-abril 2003.
Lectura interesante y divertida.
[9]Video "Scenes of software inspections. Video dramatizations for the classroom." L.Deimel.Software Engineering Institute,CMU/SEI-91-EM-5-0,1991.
[10] Thinking i Java, 3rd edition. B. Eckel. Versión electrònica a http://www.planetpdf.com/ . Versión impresa en castellano de Pearson Educación, 2002.

[5] https://testautomationu.applitools.com/junit5-tutorial/ Tutorial on JUnit.

[5] https://testautomationu.applitools.com/unit-testing/index.html Introduction on Unit Testing.

[5] https://site.mockito.org/ Library to use mock objects.

[7] https://testautomationu.applitools.com/codeless-test-automation-with-selenium-ide/ Tutorial on selenium

[7] https://testautomationu.applitools.com/cucumber-java-tutorial/ Tutorial on cucumber.

Software

JUnit (https://junit.org/junit5/)

Bugzilla (https://www.bugzilla.org/)

Jira (https://www.atlassian.com/software/jira)

Selenium (https://www.selenium.dev/)

Cucumber (https://cucumber.io/)

GitHub/GitLab (https://github.com/) (https://about.gitlab.com/)