Universitat Autònoma de Barcelona

2020/2021

Foundations of Software Engineering

Code: 102708
ECTS Credits: 3

Degree	Type	Year	Semester
2500895 Electronic Engineering for Telecommunication	OB	1	2
2500898 Telecommunication Systems Engineering	OB	1	2

The proposed teaching and assessment methodology that appear in the guide may be subject to changes as a result of the restrictions to face-to-face class attendance imposed by the health authorities.

Contact

Name: Daniel Ponsa Mussarra
Email: Daniel.Ponsa@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

Prerequisites

The subject has no prerequisites.
The subject is based on basic knowledge in C programming. Hence, it is recommended to review the C programming lessons given int the subject 'Fonaments d'Informàtica'.

Objectives and Contextualisation

The subject provides an introduction to the foundations of software engineering, to object-oriented analysis and design, and to object-oriented programming.

The basic training objectives of the subject are:

- Introducind the student in the key concepts of software engineering, seeing the singularities and differences with regard to civil engineering.
- Introducing the key concepts regarding the analysis and design of object-oriented software.
- Learning an object-oriented programming language. Specifically C ++.

Competences

Electronic Engineering for Telecommunication
- Develop personal work habits.
- Develop thinking habits.
- Learn new methods and technologies, building on basic technological knowledge, to be able to adapt to new situations.
- Work in a team.

Telecommunication Systems Engineering
- Develop personal work habits.
- Develop thinking habits.

- Learn new methods and technologies, building on basic technological knowledge, to be able to adapt to new situations.
- Work in a team.

Learning Outcomes

1. Apply the basics of software engineering to the development of software applications.
2. Critically evaluate the work done.
3. Develop independent learning strategies.
4. Develop scientific thinking.
5. Develop the capacity for analysis and synthesis.
6. Know and use the basics of programming in telecommunication networks, systems and services.
7. Manage available time and resources.
8. Manage available time and resources. Work in an organised manner.
9. Use the basics of the architecture and methodology of design, verification and validation of software.
10. Work autonomously.
11. Work cooperatively.

Content

A. INTRODUCTION TO SOFTWARE ENGINEERING
A1. Introduction: Is software engineering required? Basic Concepts

- Subject presentation.
- Software engineering vs. Civil Engineering.
- Software development cycle stages.
- Software development processes.
- Software Project planning.

A2. Analysis and Design oriented to objects

- Effective modularization of a software.
- Object-oriented design.
- UML: Class Diagram.

B. OBJECT ORIENTED PROGRAMMING IN C++
B1. Classes, Attributes, and Methods

- Declaration and definition of the basic elements.
- Class vs. Object.
- Accessibility: attributes and methods public and private.
- Function overloading

B2. Enumerates. Function call mechanism. Dynamic Memory

- Enumerates
- Function call mechanism.
- Inline features
- Passing arguments by reference / constant reference.
- Allocation /dealocation of memory.
- Pointers arithmetic.

B3. Constructors and Destructors. Operators

- Types of constructors.
- Destructors.
- Binary operators.
- The assignment operator.

- Objects and the this pointer.
- The friend concept.
- Streams. Operators >> and <<.

B4. Inheritance, Polymorphism and Genericity

- Hierarchy. Generalization vs. Specification.
- Accessibility to 'protected' attributes and methods.
- Static and dynamic binding.
- Virtual functions.
- Template functions.
- Template classes.

Methodology

The different activities that will be carried out in the subject are organized as follows:

Theory Sessions:
Part A of the curriculum (Introduction to Software Engineering) is taught in the traditional way. The basic concepts of the subject are exposed and indications are given on how to complete and deepen in this content.
Part B of the curriculum (object-oriented programming in C ++) uses a methodology based on "learning by doing", developing programs where aspects of programming previously introduced with the support of videos and associated activities are put into practice.

Problem Sessions:
In these classes, the subjects worked on in the theory classes are deepened, thus reinforcing their assimilation. Problems are solved and case studies are discussed. The proposed activities promote the capacity for analysis and synthesis, critical reasoning, and train the student in problem solving through the development of programs.

Practical work Sessions:
During the course a practice is carried out throughout different sessions. As long as the capacity of the classrooms allows it, students will work in groups of two people.
The practice will consist in the development of a C ++ program that implements a requested application.
To develop the program, the groups will work autonomously. The practice sessions will be devoted mainly to solve doubts with the teacher, and deliver milestones of the program to be developed, for further evaluation.
The groups should explain the work done to the teacher, and this will raise issues to the group members to assess the contribution of each one.

General considerations
The 'Campus virtual' platform will be used to disseminate information to students. The dates of continuous evaluation and delivery of works will be published through this medium, and may be subject to possible programming changes for reasons of adaptation to possible incidents. 'Campus virtual' will be used to inform about these possible changes, since this is the platform for the exchange of information between the teaching staff and the students.

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Exams	2.5	0.1	6, 4, 5, 8, 9
Exercise-based classes	6	0.24	1, 2, 4, 5, 10, 9
Practical work sessions	6	0.24	1, 2, 3, 5, 8, 11, 10, 9

Theoretical classes / lectures	11.5	0.46	6, 4, 5, 9
Type: Supervised			
Preparing practical assignments	15	0.6	1, 4, 7, 8, 11, 10
Resolution of problems out of the classroom	9	0.36	1, 6, 3, 5, 7, 10
Type: Autonomous			
Study	23	0.92	6, 3, 10
Tutoring and consultations	2	0.08	3, 10

Assessment

a) Processes and scheduled evaluation activities

The evaluation of the subject will be carried out continuously based on the learning evidences collected in the following processes:

- [E1]. Carrying out written tests (exams).
- [E2]. Resolution and delivery of questionnaires and exercises proposed in the sessions of theory and problems.
- [E3]. Completion of a practical work, evaluated from different activities and deliveries.

The subject consists of the following assessment activities, each evaluated with a score between 0 and 10 (inclusive):

- [E1]-ExP1, partial exam 1, 15% on the final grade.
- [E1]-ExP2, partial exam 2, 35% on the final grade.
- [E2]-Prob, exercises to be solved in the sessions of theory and problems, 15% on the final grade.
- [E3]-Prac, programming practice in C ++, 35% of the final grade.

In order to be able to pass the subject through continuous evaluation, a minimum grade of 5 must be drawn in the following 2 expressions.

- (0.3 * Grade [E1]-ExP1) + (0.7 * Grade [E1]-ExP2) + (0.1 * Grade [E2]-Prob)
- (0.15 * Grade [E1] -ExP1) + (0.35 * Grade [E1]-ExP2) + (0.15 * Grade [E2] -Prob) + (0.35 * Grade [E3]-Prac)

Keep in mind that:

- If a value inferior of 5 is obtained in the first expression, this value will be assigned as the final grade of the subject.
- Exercises in [E2]-Prob must be delivered within an established period, and will be evaluated with a score between 0 and 10 (both inclusive). Exercises not delivered within their term will be evaluated with a score of 0, and can not be recovered.
- Activity [E3]-Prac will be evaluated based on different subactivities proposed, with a delivery deadline fixed. Each sub-activity will be evaluated with a score between 0 and 10 (both inclusive). Subactivities not made or delivered out of their term will beevaluated witha note of 0, and can not be recovered.

In case of irregularities in the evaluation activities, the one detailed in section f) will be applied.

It is important to bear in mind that evaluation activities will not be carried out on a date or time different from the established one, except for justified causes, duly informed in advance to the teaching staff.

b) Programming of evaluation activities

The scheduling of the different assessment activities will be detailed in the Virtual Campus, in the Moodle classroom of the subject. The dates of the written tests (activities [E1] -ExP1 and [E1] -ExP2) will also be made public on the website of the School of Engineering, in the section of exams.

c) Recovery process

The only recoverable evaluation activities are the written tests [E1]-ExP1 and [E1]-ExP2.

The student has the option to improve the grades of these tests (one of them, or both) provided he/she has been evaluated in a set of activities that represent a minimum of two thirds of the total grade of the subject.

In order to compute the final grade of the subject, the mark obtained in the recovery exam will replace the one of the corresponding exam carried out in the continuous assessment.

In accordance with the coordination of the Degree and the direction of the School of Engineering the following activities can not be recovered:

- [E2] -Prob, 15% on the final grade.
- [E3] -Prac, 35% on the final grade.

d) Procedure for the review of qualifications

For assessment activities based on written tests ([E1]-ExP1 and [E1]-ExP2), a procedure for booking a revision date and time will be established in which the student will be able to review the activity with the teaching staff. In this context, claims can be made about the activity grade, which will be evaluated by the teachers responsible for the subject. Likewise, it is possible to arrange with the teaching staff the review of the rest of the assessment activities up to two weeks before the recovery exams.

e) Special qualifications
If the student has not performed any of the tests [E1] -ExP1 and [E1] -ExP2 the "Non-assessable" grade will be assigned. It must be remarked that according to current regulations "Non-assessable" qualifications also exhaust convocation.

In order to pass the course with honours, the final grade must be a 9.0 or higher. Because the number of students with this distinction cannot exceed 5% of the number of students enrolled in the course, this distinction will be awarded to whoever has the highest final grade. In case of a tie, partial-test results will be taken into consideration.

f) Irregularities by the student, copy and plagiarism.

Notwithstanding other disciplinary measures deemed appropriate, assessment activities will receive a zero whenever a student commits academic irregularities that may alter such assessment. Therefore, copying, plagiarizing, cheating, ... in any of the assessment activities will imply suspending it with a zero.

g) Evaluation of repeater students

From the second enrollment, repeater students may request to validate the evaluation of the activities in [E3]-Prac, taking the mark obtained the first time that the student enrolled in the subject. In order to be able to opt for this differentiated evaluation, repeater students must ask the faculty through an email.

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Evaluation test ExP1	15%	0	0	2, 6, 4, 3, 5, 7, 8, 10, 9
Evaluation test ExP2	35%	0	0	1, 2, 6, 4, 3, 5, 7, 8, 10, 9
Practical work	35%	0	0	1, 6, 3, 7, 8, 11, 10, 9

Resolution of Proposed Problems and Activities	15%	0	0	1, 2, 4, 3, 5, 10, 9

Bibliography

- R.S. Pressman, Ingeniería del software, un enfoque práctico. Mc Grah-Hill, 7a. edició , 2010.
- A. Álvarez García, R. de las Heras del Dedo, C. Lasa Gómez, Métodos Ágiles y Scrum. Anaya Multimedia, 2012.
- T.A. Pender, UML Weekeng Crash Course, Wiley Publishing, Inc. 2002.
- P. Bustamante, I. Aguinaga, M. Aybar, L. Olaizola, I. Lazacano, Aprenda C++ Básico como si estuviera en primero, Universidad de Navarra, 2004.
- P. Bustamante, I. Aguinaga, M. Aybar, L. Olaizola, I. Lazacano, Aprenda C++ Avanzado como si estuviera en primero, Universidad de Navarra, 2004.