UAB
Universitat Autònoma de Barcelona

Software Engineering

Code: 106561
ECTS Credits: 6

2025/2026

Degree	Type	Year
Artificial Intelligence	OT	3
Artificial Intelligence	OT	4

Contact

Name: Marc Castello Torrellas
Email: marc.castello@uab.cat

Teaching groups languages

You can view this information at the end of this document.

Prerequisites

Since the course is an introduction to software engineering, it will be assumed that no prior knowledge of the subject is possessed. It is the responsibility of the course itself to provide students with a means to acquire the knowledge described in the course content section (section 6 of this guide). However, it is recommended to have passed the first-year course on Fundamentals of Programming.

Objectives and Contextualisation

This course is an introduction to the software engineering process. The general objective is to provide a global and structured overview of the software development process, which goes beyond programming (which is only one phase within the entire software engineering process). The fundamental activities that make up the development process (requirements analysis, design, implementation, testing, and maintenance) will be introduced, particularly those related to Artificial Intelligence.

The aim is for students to gain a general understanding of what a software development process is, how it is modeled, which tools are used, how quality is managed, and how a project is administered. More specifically, the objectives are:

- To provide a general overview of the software development process, its paradigms, activities, and the objectives of each activity.
- To understand what software modeling is, and the importance and difficulty of requirements specification and design in its various views.
- To learn the basic concepts of UML in order to work practically on software modeling.
- To introduce students to the principles, concepts, and techniques used to manage and control software quality.
- To present the management techniques required to plan, organize, monitor, and control software projects.
- To become familiar with the SCRUM model as the most relevant paradigm of evolutionary development.

Competences

Artificial Intelligence

- Communicate effectively, both orally and in writing, adequately using the necessary communicative resources and adapting to the characteristics of the situation and the audience.
- Design, implement, analyse and validate efficient and robust algorithmic solutions to computational problems derived from the design of intelligent systems.
- Introduce changes to methods and processes in the field of knowledge in order to provide innovative responses to society's needs and demands.
- Students can apply the knowledge to their own work or vocation in a professional manner and have the powers generally demonstrated by preparing and defending arguments and solving problems within their area of study.
- Work cooperatively to achieve common objectives, assuming own responsibility and respecting the role of the different members of the team.
- Work independently, with responsibility and initiative, planning and managing time and available resources, and adapting to unforeseen situations.

Learning Outcomes

1. Analyse, design, build and maintain applications in a robust, secure and efficient way, choosing the most appropriate paradigm and programming languages.
2. Assure and evaluate the quality of a software system according to software engineering principles, standards, methodologies and practices.
3. Communicate effectively, both orally and in writing, adequately using the necessary communicative resources and adapting to the characteristics of the situation and the audience.
4. Design and evaluate human-computer interfaces that guarantee the accessibility and usability of computer systems, services and applications.
5. Propose new ways of measuring the success or failure of implementing innovative proposals or ideas.
6. Students can apply the knowledge to their own work or vocation in a professional manner and have the powers generally demonstrated by preparing and defending arguments and solving problems within their area of study.
7. Understand and apply software engineering principles, methodologies and life cycles.
8. Weigh up the risks and opportunities of both your own and others' proposals for improvement.
9. Work cooperatively to achieve common objectives, assuming own responsibility and respecting the role of the different members of the team.
10. Work independently, with responsibility and initiative, planning and managing time and available resources, and adapting to unforeseen situations.

Content

BLOCK 1. SOFTWARE AND SOFTWARE ENGINEERING

Topic 1. Principles of Software Engineering

- Definition and objectives of SE: Definition, characteristics, software applications. Definition of SE and objectives of SE.
- Software evolution: Stages, software crisis, problems and causes.
- Process, method, and tool: Definitions, activities in the software development process.
- Software development paradigms: Linear sequential model (classic life cycle), prototyping model, evolutionary model, and spiral model.

BLOCK 2. SOFTWARE PROJECT MANAGEMENT AND DEVELOPMENT

Topic 2. SCRUM

- Introduction, Agile project management methodology.
- SCRUM: Characteristics and history.
- SCRUM components: Roles, artifacts, processes, scalability, and Scrum of Scrums.
- Software tools.
- Other agile models: Lean, Extreme Programming, Kanban.
- A case study: Building a city with Lego using Scrum.

BLOCK 3. SOFTWARE MODELING

Topic 3. Software Requirements Analysis

- Introduction: Types of requirements, tasks to be carried out.
- Understanding the problem: Communication techniques, associated problems, principles of analysis.
- Requirements specification: Desirable properties of an RS, SE standards, review and validation of the specification.

Topic 4. Software Design

- Introduction: Design process, data design, architectural design, interface design, procedural design, and design principles (objectives).
- Design concepts: Abstraction, modularity, and refinement.
- Effective modular design: Functional independence, cohesion, coupling, heuristics for effective modular design.
- User Interface Design.

Topic 5. UML (Unified Modeling Language)

- Evolution and history of UML.
- UML views: Use case view, logical view, component view, and deployment view.

BLOCK 4. SOFTWARE QUALITY MANAGEMENT

Topic 6. Software Quality

- Introduction: Quality concepts.
- Software testing: Testing strategies, testing of conventional applications, testing of web applications.
- Software configuration management.

Activities and Methodology

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Preparation and Study	30	1.2	1, 7
Seminars/Problems	14	0.56	3, 4, 8, 5, 6, 9
Theory Classes	30	1.2	1, 2, 7
Type: Supervised			

Interviews/Tutorials for Case Study	15	0.6	1, 2, 3, 7, 6, 9, 10
Type: Autonomous			
Practical Work	38	1.52	1, 2, 3, 7, 4, 8, 6, 9, 10
Tutoring	15	0.6	1, 2, 7

The Software Engineering course has a strong applied component, but it also requires a theoretical foundation, based on experience, from which practical decisions are implemented. Therefore, the course will not follow a purely PBL (Problem-Based Learning) approach, but rather a combination of PBL and traditional methodology. Students will be presented with a practical case (a generic problem) of real-world scale, around which problem-solving sessions and practical work will be structured. The student must assume the role of software analyst/architect in order to make the appropriate decisions leading to the construction of a design proposal for the proposed system. The instructor will take on the role of tutor and, in certain situations, act as a client, encouraging developer-client dialogue simulations.

The course consists of 4 weekly in-person hours. No formal distinction is made between lecture, problem-solving, and lab sessions. During class hours, lectures, problem-solving sessions, or practical work will alternate according to a schedule that will be provided on the first day. In general, the fundamental theoretical content will be covered during the first sessions of the course (approximately the first 6 weeks), but some problem-solving and case-based sessions will also be progressively included. Once the theoretical content has been delivered, all remaining sessions will be dedicated to problem-based work, with certain mandatory attendance sessions where practical work will take place (in teams of *n* people, with *n* defined at the beginning of the course) or problems will be solved and submitted at the end of the session.

THEORY

Consists of lectures with multimedia material available on the UAB Virtual Campus. The main goal of these lectures is to introduce the basic concepts of the processes, methods, and tools related to the software life cycle from an engineering perspective. Principles and paradigms of Software Engineering, the various activities, and the tasks to be carried out in each activity will be presented. These theory classes are intended to provide students with a realistic view of the rigor, planning, and systematic approach required in professional software development.

SEMINARS / PROBLEM SESSIONS

Seminars are participatory sessions involving student engagement. They serve a dual purpose: on one hand, linking with the lectures, and on the other, with the practical sessions related to the problem to be solved in teams. Seminars will complement the technical content presented in the lectures, supporting a deeper understanding. At the same time, they will provide the framework in which the guide problem is discussed, its foundations are established, and development is carried out. Students will be given the necessary knowledge to advance in the practical work or will be guided on how to acquire it. Seminars primarily foster analytical and synthesis skills, as well as critical reasoning and decision-making by the students when addressing the guide problem. Since this is a very practical course and the theoretical content is relatively straightforward, practical work is often reinforced with problem-solving or participatory activities. Seminar content will be coordinated with the stages of the practical project.

PRACTICAL WORK MONITORING

There will be no dedicated in-person lab sessions. Students will solve the guide problem autonomously insmall groups, with progress monitored during the regular class schedule. The practical sessions (12 hours over the course) will consist of two parts. First, each practical team will present their project progress to the rest of the class (the presentation will be done by the team member acting as Scrum Master for that iteration). In the second part of the session, the practical tutor will hold an individual interview with each team to monitor and evaluate their progress. Development will follow a SCRUM-based approach, where each increment, or *sprint*, corresponds to a session.

Attendance to the follow-up interviews is MANDATORY for all students. Absences will negatively affect the evaluation of the practical work, and failure to attend more than 15% of the sessions will result in not passing this component. In case of justified absence, instructors may provide compensation mechanisms. Team sizes will be determined at the beginning of the course, and as a guideline, teams will consist of 5 members. Since attendance to follow-up sessions is mandatory and takes place during class hours, it is important that all team members are enrolled in and attend the same group.

PRACTICAL PROJECT

At the beginning of the course, students will be given a guide problem. This problem serves two purposes. First, it will be used as an example in seminar sessions, where relevant parts will be addressed in line with the course content. Second, it functions as the specification for a practical assignment that will be progressively solved in stages throughout the semester. Students will organize themselves into work teams. The project will be carried out autonomously, with the weekly follow-up process described above.

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.

Assessment

Continous Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Practical Evaluation	60%	4	0.16	1, 2, 3, 4, 8, 5, 6, 9, 10
Theoretical Evaluation	40%	4	0.16	2, 7, 6

INDICATORS AND ASSESSMENT

This course does not offer the single-assessment system.

Assessment will be based on the practical work developed by the student and included in their portfolio (final resolution of the guide problem, material generated in seminar activities), the degree of involvement in seminars, and the technological knowledge acquired in the subject. The following assessment instruments will be considered:

- Theoretical Assessment (TA). An in-person written test to evaluate the degree of knowledge acquired by the student on an individual basis. Two midterm tests will be held during the course that will allow students to exempt content, provided the grade is equal to or higher than 4. Each midterm will account for 50% of the final theoretical assessment grade.

- Practical Assessment (PA). Evaluation of the overall performance of the different working groups in solving the guide problem (project) and the various activities presented in seminars (problems). It consists of two components:

  Project Grade
  Documentation submitted by the students about their practical work. The technical resolution of the guide problem will be assessed, as well as the defense of the work during submissions, monitoring of work progress, and active participation in seminars.
  Assessment will be based on interviews with the professor, short written tests, submissions, or presentations of problems and case studies presented during the activities. Although this instrument is collective, at the tutor's discretion, it may be graded individually in cases where a student's participation stands out (positively or negatively) compared to their group.

Problem Grade
During the course, the professor will ask students to solve some problems independently, in addition to those solved jointly during problem sessions. These submissions are individual.

The final grade will be calculated according to the following table:

Grade	Minimum Grade	Average Grade	Minimum Grade	Average Grade
Midterm 1	4	TA=0.5 Midterm_1+0.5 Midterm_2	5	FG = 0.4 TA + 0.6 PA
Midterm 2	4			
Problems	5	PA=0.5 Problems+0.5 Project	5	
Project	5			

At the time of publishing each evaluation, a review period will be established. These review periods will be strictly respected, and no grades will be reviewed outside of the official review period.

The course will be graded as "Not Assessable" only if the student has not attended any of the evaluation activities and has not submitted the assigned work (in full or partially).

If the student does not pass the course because any of the evaluation activities do not reach the required minimum grade, the final grade recorded will be the lower value between 4.5 and the weighted average of all marks.
Exceptions:

- A "Not Assessable" grade will be given to students who do not participate in any evaluation activity.
- A grade of zero (0) will be given in the event of irregularities during an assessment activity, in which case passing by compensation will not be possible.

Honours distinctions will be awarded according to UAB regulations (depending on the number of enrolled students) to the highest grades equal to or above 9.

For each evaluation activity, the place, date, and time of the review will be indicated so that students may review the activity with the instructor. In this context, claims regarding the grade can be submitted and will be evaluated by the responsible teaching staff. If the student does not attend the review session, the activity will not be reviewed later.

See the "PLAGIARISM" section for measures related to irregularities in evaluation activities.

RECOVERIES

- Theoretical Assessment (TA): Two midterm theory exams will be conducted during class time. Students who do not pass one or both tests will have a chance to recover via an exam scheduled on the official final evaluation date of the program.
- Practical Assessment (PA): The practical work is assessed through continuous evaluation during follow-up sessions. Therefore, there will be no recovery activity at the end of the course. However, regarding project resolution, failure to submit or pass an intermediate delivery may be recovered by

rescheduling the project plan for later sessions during the course, subject to agreement with the instructor.

ASSESSMENT DATES

Thedates for continuous assessment and work submissions will be published on the virtual campus and may be subject to changes due to unforeseen circumstances. All changes will be communicated through the virtual campus, which is the standard channel for information exchange between teacher and students.

REPEATING STUDENTS

Partial grades (theory or practice) from previous years are not retained. However, at the discretion of the instructor and based on prior assessments, some compensations may be granted.

PLAGIARISM

Without prejudice to any disciplinary measures deemed appropriate, and in accordance with current academic regulations, any irregularities committed by a student that may affect grading will result in a zero (0).
For example, plagiarizing, copying, allowing copying, etc., will result in failing that assessment activity with a zero (0).
These activities will not be recoverable.
If passing such an activity is necessary to pass the course, the entire course will be failed, with a grade of zero (0), and no opportunity to recover it within the same academic year.

Such irregularities include, among others:

- Total or partial copying of a practice, report, or any other assessed activity.
- Allowing others to copy.
- Submitting a group project not entirely completed by all group members.
- Presenting material created by a third party as one's own, including translations or adaptations, or any work with non-original content.
- Having communication devices (e.g., mobile phones, smartwatches, etc.) accessible during individual theoretical-practical exams.

A student who commits irregularities during an assessment will not have any of their other evaluation activities validated.
In short: copying, allowing to copy, or plagiarizing (or attempting to) in any assessment activity equals a FAIL with a zero (0), non-compensable, and not valid for future course recognition.

USE OF ARTIFICIAL INTELLIGENCE (AI)

The use of AI technologies as part of the project development is permitted as long as the final result reflects a significant personal contribution by the student in the analysis and reflection process.
Students must clearly indicate which parts were generated using AI, specify the tools used, and include a critical reflection on how these tools influenced the process and the final outcome.
Lack of transparency in the use of AI will be considered academic dishonesty and may result in grade penalties or more severe sanctions in serious cases.

FINAL CLARIFICATION

In case of doubt or discrepancy, the most updated information communicated on the course presentation day and published on the virtual campus will prevail.

Bibliography

G. Booch, J. Rumbaugh, I. Jacobson. El lenguaje unificado de modelado. Addison-Wesley, 2001.

I. Jacobson, G. Booch, J. Rumbaugh. El proceso unificado de desarrollo de software. Addison-Wesley, 2000.

J. Rumbaugh , I. Jacobson, G. Booch. El lenguaje unificado de modelado: manual de referencia. Addison-Wesley, 2007.

T. Quatrani. Visual Modeling with Rational Rose 2000 and UML. Addison-Wesley, 2000.

P. Krutchen. The Rational Unified Process. An Introduction. Addison-Wesley, 2000.

Roger 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.

K. Pohl, C. Rupp. Requeriments Engineering Fundamentals. Rocky Nook Inc. 2011.

Atès que avui en dia molts materials estan en línia, molt més actualitzats que les fonts bibliogràfiques, durant el curs es proporcionen enllaços a documentació lliure a la xarxa,video-lectures, etc.

Software

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

Miro (https://miro.com/)

Figma (https://www.figma.com/)

Balsamiq (https://balsamiq.com/)

Groups and Languages

Please note that this information is provisional until 30 November 2025. You can check it through this link. To consult the language you will need to enter the CODE of the subject.

Name	Group	Language	Semester	Turn
(PAUL) Classroom practices	711	English	second semester	morning-mixed
(TE) Theory	71	English	second semester	morning-mixed