UCLouvain

linfo2255

2026

Software engineering project

6.00 credits	30.0 h + 30.0 h	Q1

Language :	English > French-friendly
Place of the course	Louvain-la-Neuve
Prerequisites	Required#: initial experience in computer projects, as targeted in course LINFO1001 Required#: skills in C programming language as taught in course LEPL1503
Main themes	This course treats a specific *advanced topic* or selection of topics of *current research interest* in the area of software engineering. The actual topic(s) may vary from year to year, and will be chosen from a variety of software engineering domains such as data-intensive computing, software analytics, development and analysis of large evolving software systems, big data techniques, software repository mining, software recommendation systems, software visualization, novel programming technologies, software requirements and analysis,model-driven software engineering, software configuration management, software engineering processes, software engineering tools and methods, software testing and quality aspects, etc.
Learning outcomes	**At the end of this learning unit, the student is able to :** Given the learning outcomes of the "Master in Computer Science and Engineering" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes: · INFO2.1-5 · INFO4.1-4 · INFO5.1-6 · INFO6.1, INFO6.5 Given the learning outcomes of the "Master [120] in Computer Science" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes: · SINF1.M3 · SINF2.1-4 · SINF4.1-4 · SINF5.1-3 · SINF6.2-4 Given the learning outcomes of the "Master [60] in Computer Science" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes: · 1SINF1.M3 · 1SINF2.1-4 · 1SINF3.1-4 · 1SINF4.1-3 · 1SINF5.2-4 At the outcome of this course, the students will have acquired the necessary competences to build a large-scale software system under semi-professional working conditions. More specifically, students having completing this course with success will be able to: · describe the differences among several major process models (e.g., waterfall, iterative, and agile); · differentiate among the phases of software development (specification, architecture, design, implementation, validation, documentation); · complete, in a rigorous and systematic way, the artefacts produced in these different software life cycle phases; · apply a software development methodology currently practiced in industry; · work efficiently in a team to develop a medium-to large-scale software system; · manage the coordination and communication between the different team members; · interact with a client to identify his requirements, to clarify imprecise specifications, and to take into account requested modifications throughout the development process; · describe the functional requirements of a software system using, for example, use cases or users stories; · estimate the time and resources needed to complete such a software development project, plan the tasks to be executed and the deliverables to be produced, and respect this planning; · use some project management tool to assign and follow the planned software development tasks; · put in practice different methods and techniques to assure the quality of the produced software; · understand the problems inherent to the development of large software systems having different stakeholders and that consist of multiple components.

Evaluation methods	Assessment of the course will be based on : · Individual, continuous, active and balanced participation in group work and weekly meetings with course assistants/tutors; · The completion of two to three intermediate prototypes, accompanied by their corresponding technical reports; · The final report, the delivered system and its documentation, as well as the presentation and demonstration of the final product to the customer or an intermediary; · The integration of the solutions from the various groups into a complete product. **Important!** As this course is based on participation in a team project throughout the year, project grades will automatically be kept in the second session. It will therefore not be possible to redo the project or improve your grade in the second session. **Important note** : The active and balanced participation of each student is expected in all aspects of the project: · analysis and design, · development (implementation and testing), · integration, · documentation (reports and demonstration). This distribution aims to ensure a complete and equitable learning experience for all. For each prototype, a student must demonstrate a tangible contribution in each of these areas. If a student fails to participate in one or more of these aspects, he or she may be attributed an individual mark of 0 for the corresponding prototype. **Reminder** : Failure to comply with the rules concerning the use of generative AI (see Teaching methods) may result in penalties up to and including the attribution of a mark of 0 for the student or group of students involved in the activity concerned.
Teaching methods	The course is based on the complete development (analysis, design, implementation, validation, documentation, integration and deployment) of large-scale software for a potential client, in teams of 6 to 8 students, supervised by a project leader. Weekly meetings will be held with the project leader to follow up the work, plan tasks and monitor the progress of the project. Various prototypes and technical reports will be produced over the course of the semester. Particular attention will be paid to the responsible use of generative artificial intelligence tools: · They can only be used as tools to support reflection in the initial analysis phase, for example to stimulate brainstorming and initiate discussions on the identification of needs and requirements. · On the other hand, they may not be used to produce code (implementation and testing) or to generate design artefacts. · This rule, which will be strictly applied, aims to preserve the authenticity of the students' software engineering work, while allowing them to critically explore the role of emerging technologies in the software development process.
Content	This software engineering project consists of the development (analysis, design, implementation, validation, documentation, integration and deployment) of a realistic and non-trivial software application, if possible proposed by and with the participation of a real client, under semi-professional working conditions. The topic of the application to be constructed is proposed by an industrial partner or a non-profit organization that participates in the organisation of this course. Teams of 6 to 8 students (required to achieve such a large project), will collaborate, supervised by a project manager. Weekly meetings will be held with the project manager (an assistant or tutor) to present the progress and difficulties encountered, to evaluate alternatives, and discuss the distribution and planning of the work among team members. The application to be developed will most likely be a web application, but the choice of the programming language, the environment, the application framework, and the development tools will depend on the requirements of the project client.
Inline resources	http://moodleucl.uclouvain.be/course/view.php?id=7599
Bibliography	**French** Des lectures supple#mentaires seront sugge#re#es dans le plan de cours qui de#crit les produits livrables et l'organisation du projet. Les supports de cours pertinents, des slides et des informations pratiques seront accessibles sur Moodle, qui sera e#galement le principal moyen de communication entre l'enseignant et les e#tudiants. **English** Additional reading material will be suggested in the course plan which describes the deliverables and organisation of the project. All relevant course material, slides and practical information will be available on Moodle, which will also be the main means of communication between the teacher and the students.

Other infos	Prerequisites: · Have good knowledge of and experience with the concepts of object-oriented programming, algorithms and data structures. · Have participated in the development of a small to medium-sized software system.
Faculty or entity in charge	INFO

Programmes containing this learning unit (UE)

Program title	Acronym	Credits	Prerequisite	Learning outcomes
Master [120] in Computer Science and Engineering	INFO2M	6		🔍
Master [120] in Computer Science	SINF2M	6		🔍
Master [60] in Computer Science	SINF2M1	6		🔍