Exchange programme Vrije Universiteit

Vrije Universiteit Amsterdam - Exchange programme Vrije Universiteit - 2022-2023

Exchange

Vrije Universiteit Amsterdam offers many English-taught courses in a variety of subjects, ranging from arts & culture and social sciences, neurosciences and computer science, to economics and business administration.

The International Office is responsible for course approval and course registration for exchange students. For details about course registration, requirements, credits, semesters and so on, please visit the exchange programmes webpages.

Software Design

Course Code	XB_40007
Credits	6.00
Period	P4
Course Level	200
Language Of Tuition	English
Faculty	Faculty of Science
Course Coordinator	dr. I. Malavolta
Examiner	dr. I. Malavolta
Teaching Staff	dr. I. Malavolta
Teaching method(s)	Lecture, Seminar

Course Objective

The main objective of the course is to let you master model-based design methodologies and techniques, obtain insights and knowledge about recurrent software design problems and object-orientation. (Knowledge and understanding)
In addition, you will develop critical reasoning skills to select the most appropriate object-oriented design patterns and apply them to the (software) problem at hand. (Making judgements) (Applying knowledge and understanding) (Lifelong learning skills)

Course Content

Developing real software systems is complex; they are large, and their development often starts when it is still unclear what they should precisely do. The goal of software design is to model modern, complex software systems in a systematic manner. The lectures will cover and apply a number of software modeling techniques. You will learn which technique is the most appropriate for which problem, how to describe a (software) problem in models, how to use such models to reason about software, and finally how to use models to discuss your ideas and plans with other stakeholders so that requirements are clarified and software systems are well understood and developed in a more reliable way. The course also introduces the most known design patterns for creating robust, better organized, and maintainable software systems. Design patterns can be considered as standardized methods of solving recurrent design problems regarding object-oriented software systems.The course is based on the Unified Modelling Language (UML).

Additional Information Teaching Methods

Lectures (H). Modeling exercises (W). Weekly presentations (pre).

Method of Assessment

In this course the assessment is composed of two components:

- Team project (70% of the final grade): it will be carried out throughout the whole course by groups of students; each part of the project will be started during the laboratory sessions and finished as homework so that you will likely be on track within the course schedule. The result of the team project is composed of two parts: (i) a modelling part, and (ii) an implementation part. Both the modeling and implementation parts will be evaluated by the instructors according to a shared assessment rubric. Within the team project, each of you will be responsible for a certain part of the project; as a team, you will

report the responsibilities that each team member took in the project. In case of issues in some specific parts of the project, the team member responsible for them will be the contact point for them.

- Written exam (30% of the final grade): it consists of 20 multiple-choice questions (20 points).
The exam aims to assess your knowledge of the methodologies, constructs of the UML language, practical insights that are discussed in the lectures. The written exam is based on the mandatory books listed in the Readings section of this guide.

- Code review (pass/fail): for assignment 2 each student will perform a code review on the source code of the project of another team. Instructions on how to do code reviews will be provided during the course. Code reviews will benefit both (i) the reviewer since they will have the chance to look at other teams' code and get inspiration from them and (ii) the team receiving the review since they will get additional feedback about how to improve their project.

Literature

- [Main text book] Martina Seidl, Marion Scholz, Christian Huemer, Gerti Kappel, "UML@Classroom: An Introduction to Object-Oriented Modeling", 2015.
- John Ousterhout, “A philosophy of software design”, Yaknyam Press, 2018.
- Martin P. Robillard, “Introduction to Software Design with Java”, Springer, 2019.
- [only Section 3.4 and Chapter 8] Ian Sommerville, “Engineering Software Products”, Pearson, 2019

Additional Information Target Audience

Bachelor Computer Science (year 2)

Recommended background knowledge

Object-oriented principles in any programming language (for instance Scala, Java, C/C++, Python).