UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

COURSE SYLLABUS

Software Engineering

2526-3-E3101Q119

Aims

Acquire more advanced knowledge of software development than that acquired during the II year course of software analysis and design. Know and apply architectural patterns during software development. Identify and remove code violations through the use and support provided by some tools such as SonarQube and SonarCloud. Introduction to DevOps and Continuous Integration.

Knowledge and Understanding
Upon completion of this course, the student will have acquired solid knowledge and skills in the development of complex software projects, use of architectural patterns, and on code quality assessment.

Applied Knowledge and Ability to Understand
The student will be able to apply all that he/she has learned in the 2nd year Software Analysis and Design course and the 3rd year Software Engineering course to the analysis and development of a complex project, in the application of design patterns and architectural patterns, in the evaluation of the quality of the software produced through metrics and recognition of anomalies and problems in the software and their resolution. Each student will be expected to know and use tools well known in companies for code quality assessment.

Autonomy of judgment
The course promotes the development of critical thinking through comparative evaluation of design and methodological solutions, through comparison of tools and through individual contribution in the examination project. Students will also have the opportunity to attend one or two company seminars on topics of great interest to software engineering, which will further stimulate judgment skills.

Communication Skills
The student will be able to communicate effectively through an individual presentation of an architectural pattern in the classroom, the development of the examination project, the document produced on the examination project, and the individual oral examination.

Learning skills

The course promotes the development of autonomy in study through an active teaching approach. At the end of the course, the student will be able to independently pursue advanced software engineering topics, which may also result in an internship/thesis, consolidating the basis for continued learning beyond the curricular context.

Contents

Principles, techniques and tools for software development. Architectural patterns and examples of their application in software development. Best practices in Java. Code quality evaluation through SonarQube. Examples of software projects and discussion on the issues addressed during the course

Detailed program

1 Presentation of the course. Objectives and contents.
Software Engineering: Introduction to Model-Driven Software Engineering, Component-Based Software Engineering, Service-Oriented Software Engineering, Distributed Software Engineering.
2 Application of design patterns in software development. Modeling a persistence framework with design patterns.
3 Software architectures. Software architecture design. Architectural patterns for enterprise applications. Package, component and deployment diagrams.
4 Best Practices in Java. Reflexivity in Java.
5 Self-managed and self-adaptive systems: fundamental concepts, application domains, and case studies. Model-driven engineering.
6 Service-oreinted software engineering: fundamental concepts. Migration to microservices.
7 Software Quality Assessment. Software evaluation metrics. Design principles of Martin.
8 . dentification and removal of code violations through SonarQube (Lab).
9. Use of Git and GitHub, team cooperation during the development of a project (Lab).
10. Software project management: basic concepts. Project planning: Gantt charts. Risk management, quality management.
11 . Use of Sonercloud (Lab)
12. Use of the Undersand tool.
13. Introduction to DevOps and Continuous Integration: using Github Actions

Prerequisites

Object-oriented analysis and design.

Programming in Java.

Teaching form

Lessons in presence and in Italian language.

Lessons with slides in Italian or in English.

6 lessons of 2 hours in presence
10 lessons of 2 hours in presence with students interactions
10 activities of 2 hours with exercises and students interactions.
8 laboratory activities of 3 hours in presence with students interactions

Textbook and teaching resource

Sommerville, Ingegneria del Software, Pearson, 8° ed, 2007.

C. Larman, Applicare UML e i Pattern – analisi e progettazione orientata agli oggetti, Pearson, 3° ed, 2005.

Most of the material to prepare the exam will be available on line.

Semester

I semester

Assessment method

Development of a complete project in a group of 3-4 students though also the exploitation of different tools SonarQube, Understand. Evaluation in the range 0-22.

Oral examination. Evaluation in the range 0-5.

Lab activity evaluation. Evaluation in the range 0-4.

Task assigned to each student during lessons. Evaluation in the range 0-2.

Office hours

By appointment through email

Sustainable Development Goals