COMPUTER ENGINEERING (LM55)

(Lecce - Università degli Studi)

Teaching SOFTWARE ENGINEERING

Teaching in italian SOFTWARE ENGINEERING

Teaching SOFTWARE ENGINEERING

SSD code ING-INF/05

Reference course COMPUTER ENGINEERING

Course type Laurea Magistrale

Credits 9.0

Teaching hours Front activity hours: 81.0

For enrolled in 2019/2020

Taught in 2019/2020

Course year 1

Language ENGLISH

Curriculum PERCORSO COMUNE

Location Lecce

Semester Second Semester

Exam type Oral

Assessment Final grade

Course timetable https://easyroom.unisalento.it/Orario

GenCod A003074

Owner professor LUCA MAINETTI

BRIEF COURSE DESCRIPTION

After the course the student should be able to:

- Apply main software engineering principles and control software qualities (both internal and external);
- Design and implement software following industrial standards (UML) and structured software production processes;
- Manage the software engineering i.e. execute tasks as planning, organizing, staffing, controlling, estimating (software cost and size);
- Design the software adopting standard software architectures;
- Select and adopt software design patterns (creational patterns, structural patterns, behavioral patterns);
- Verify the software exploiting standard tools and adopting well-known metrics;
- Develop complex model-view-controller web and mobile software systems, exploiting at the back end the Spring framework, and at the front end the AngularJS framework, connecting them through RSRT/JSON web services;
- Use the main open source tools for the software testing and refactoring, and for the software configuration management.

REQUIREMENTS

The prerequisites for attending the corse are the knowledge of structured programming languages (Java) and the fundamentals of computer science.

COURSE AIMS

The main goal of the course is to deepen students' knowledge on modern design and development techniques for interactive software systems. In particular, methods and tools for automated software testing, agile processes organization and design patterns selection will be analyzed. All concepts will be experimented by students designing, developing and testing a software prototype of a service based web application with a mobile extension (app). The software prototype will be developed on top of modern frameworks (Spring, Angular, Ionic).

TEACHING METHODOLOGY	Classroom lessons, classroom practice, project work in pair programming.
ASSESSMENT TYPE	The exam consists of two tests: a written test, intended to verify the theory of software engineering concepts (10 points out of 30); a software prototype implementation, intended to verify the practice of design patterns, MVC architectures and tests, which will be discussed during an oral examination (20 points of 30). Both written test and software prototype implementation are mandatory. The software prototype should be developed in pairs. The software system must be designed using UML, adopting standard design patterns. The software system must be developed starting from MVC frameworks (Spring, AngularJS), using a structured programming language, and must be systematically tested collecting metrics. A mobile extension of the software system is required. The software prototype must be developed following an agile process and must be documented. A month before the end of the course, the general requirements of the software prototype will be published by the teacher, a new requirements set for each year. The requirements will be effective till a new set of specifications will appear. The mark of the written exam has the same temporal extension of the project's requirements.
ASSESSMENT SESSIONS	See www.ing.unisalento.it.
OTHER USEFUL INFORMATION	Master's degree in Computer Engineering, Department of Innovation Engineering, teaching sector ING-INF/05, 9 CFU, 74 hours in classroom (both theory and practice), 54 hours of project work in pair, 1° year, 2° term. **To meet the teacher**: Monday form 3:00 PM to 7:00 PM at the teacher office, La Stecca building, 1° floor, central staircase, Ecotekne campus.

FULL SYLLABUS

- Software engineering principles (8 hours):
  - Software qualities and software engineering principles;
  - Software production process;
  - Management of software engineering.
- Software architectures (4 hours):
  - Design and software architectures;
  - Software architectures specification.
- Software design pattern (12 hours):
  - Introduction to standard architectures and design patterns;
  - How to select and adopt a design pattern;
  - Creational patterns, structural patterns, behavioral patterns.
- Software verification (10 hours):
  - Introduction to man software verification methods;
  - Black-box and white-box methods;
  - Test in the large, test in the small, correctness proofs;
  - Software metrics.
- Introduction to Spring framework (14 hours):
  - Introduction to Java EE;
  - Creating a dynamic web project with Java EE;
  - Introduction to Spring framework and development environment setup;
  - Developing a Spring MVC application;
  - Accessing Data Layer with Spring Data JPA;
  - Building a RESTful Web Service.
- Software development and verification tools (5 hours):
  - JUnit;
  - Refactoring;
  - Concurrent Versioning Systems (GIT).
- Cloud computing with Amazon Web Services and EC2 (7 hours):
  - Introduction to Amazon EC2 platform;
  - Introduction to Amazon API Gateway;
  - Configuring an EC2 instance and publishing API.
- Mobile apps development with AngularJS (14 hours):
  - Angular: Project Setup;
  - Angular: Component, Template & Data Binding;
  - Angular: Forms (Input, Validation, Template-Driven);
  - Angular: Services, Routing, HTTP;
  - Developing Cross-platform Mobile App with Ionic.

REFERENCE TEXT BOOKS

1. Ghezzi, Jazayeri, Mandrioli - Fundamentals of Software Engineering (2nd edition) - Pearson College Div 2002.
2. Fowler - UML Distilled (3rd edition) - Addison Wesley Object Technology 2003.
3. Gamma, Helm, Johnson, Vlissides - Design patterns - Addison Wesley 2002.
4. Larman - Agile and Iterative Development: A Manager's Guide - Addison-Wesley Professional 2003.
5. Beck - Test Driven Development: By Example - Addison-Wesley Professional 2002.
6. Fowler, Beck, Brant, Opdyke, Roberts - Refactoring: Improving the Design of Existing Code - Addison-Wesley Professional 1999.
7. A Java advanced programming reference guide.