Software foundations

Number of ECTS Credits

3

Abbreviation

TSM_SoftFound

Version

2016.03.17

Responsible of module

Tiziano Leidi, SUPSI

Language

	Lausanne	Bern	Zürich	Lugano/Manno
Instruction	☐ E ☐ F	☐ D ☐ E ☐ F	☐ D ☐ E	☒ E
Documentation	☐ E ☐ F	☐ D ☐ E ☐ F	☐ D ☐ E	☒ E
Examination	☐ E ☐ F	☐ D ☐ E ☐ F	☐ D ☐ E	☒ E

Module category

☐ Fundamental theoretical principles

☒ Technical/scientific specialization module

☐ Context module

Lessons

☒ 2 lecture periods and 1 tutorial period per week

☐ 2 lecture periods per week

Brief course description of module objectives and content

The goal of the module is to provide software-design competences covering the software development process, software modeling and object-oriented programming

Aims, content, methods

Learning objectives and acquired competencies

- to understand and be able to employ the software development process
- to learn and understand the basic solutions for software modeling
- to learn the object-oriented principles and the best practices for software programming
- to be able to manage the software-development lifecycle from requirement analysis to the testing phase

Contents of module with emphasis on teaching content

Software modeling

An overview of the software development process will be given by explaining both of its two main activities:

1. The contents realization activity: focus on methods and techniques that allow solving a specific problem by analyzing it in its domain. These permits to obtain a conceptual model that could be finally transformed into and executable implementation.
2. The management activity: concentrate on organizing and managing resources (people, money, time, communication, quality, risk, etc.), in a way that these resources deliver all the work required within the defined scope and constraints.

This introduction will focus on:

- life-cycle strategies: linear models, iterative models, evolutionary models
- requirements engineering with focus on "use cases" methodology
- metrics and testing: unity, integration, validation and system testing
- principles of software development with focus on OO methodologies
- Unified Modeling Language (UML) with examples of its widely adopted diagrams, like the class diagram and the sequence diagram
- principles of architecture design, from main design to subsystems

Object Oriented Programming

Object-oriented programming (OOP) is a programming methodology that concentrates on the concept of "classes of objects" to design computer programs. It focuses on several software development paradigms that includes: abstraction, inheritance, polymorphism, and encapsulation.

The most fundamental principles of object-oriented programming will be introduced by highlighting the importance of the idea of "classes of objects" in the design of software applications. Advantages of OOP compared to procedural programming will be illustrated by showing some practical examples. The JAVA programming language will be used in the examples and exercises. This introduction will focus on:

- Java fundamentals with an overview of its syntax and main concepts

- OO principles and structures with focus on "classes of objects"
- Development best practices with hints for the realization of programs

Teaching and learning methods

Frontal theoretical lessons, demonstrations, examples, exercises

Prerequisites, previous knowledge, entrance competencies

Skills on procedural programming

Literature

Assessment

Certification requirements for final examinations (conditions for attestation)

Written module examination

Duration of exam: 120 minutes
Permissible aids: 2 pages (A4) with summary