MASTER OF SCIENCE
IN ENGINEERING

Module Description

Software Engineering and Architectures

General Information

Number of ECTS Credits

3

Abbreviation

TSM_SoftwEng

Version

11.6.2017

Responsible of module

Martin Kropp

Language

	Lausanne	Bern	Zürich
Instruction	☐ E ☒ F	☐ D ☐ E ☐ F	☒ D ☐ E
Documentation	☒ E ☐ F	☐ D ☐ E ☐ F	☐ D ☒ E
Examination	☐ E ☒ F	☐ D ☐ E ☐ F	☒ D ☐ 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 module provides an in-depth view of selected topics of modern software engineering. These stem from the fields: modern software development processes, software architecture, and the principles of evolution of software systems.

Learning objectives and acquired competencies

- The student applies and understands benefits and liabilities of agile and lean development
- The student knows about advanced architectural and design patterns and uses them to drive and reflect on design decisions
- The student has the awareness of software as a continuously evolving and complex system
- The student knows and can select maintenance and evolution techniques for continuous development of evolving and extension of legacy software systems while maintaining its quality

Contents of module with emphasis on teaching content

Modern Software Engineering

- Agile Development
    - value creation
    - risk management
    - team culture
    - customer relations
- Mechanisms and Practices
    - effective communication among stakeholders
    - project retrospectives, feedback techniques
    - quality management
    - change
    - ongoing requirements, solicitation and management
    - incremental planning
- Modern Software Engineering Methodologies
    - overview and comparison of different approaches e.g. XP, Scrum, Kanban
    - implications for project management

Software Architectures

- Role of software architecture and software architects
    - reference models, reference architectures
    - architectural structures and views
    - software architecture documentation
- Advanced design concepts
    - the SOLID principles
    - Attribute Driven Design
    - design by contract
- o Architectural patterns
    - for distributed architectures
    - architecture patterns vs design patterns
    - pattern styles
- Selection, creation, and evaluation of software architectures
    - quality attributes
    - architecture analysis methods

Software Evolution

- Principles of Software Evolution
    - development, maintenance, evolution
    - software aging
    - Program comprehension
- Software Quality & Analysis
    - software quality metrics
    - software analysis and visualization
    - continuous quality control
- Evolution of Legacy Code
    - “Re”-Techniques: Reverse Engineering, Re-Engineering, Re-Factoring
    - object-oriented re-engineering
    - working effectively with legacy code
    - Testing legacy systems

Teaching and learning methods

Weekly 2-hour lecture with 1 hour of exercises

Self-study, homework assignments

Prerequisites, previous knowledge, entrance competencies

- Object-oriented programming and design in more than one programming language
- Unified Modeling Language (UML)
- *Design Patterns: Elements of Reusable Object-Oriented Software* (Gamma, Helm, Johnson, Vlissides; ISBN 0-201-63361-2)
- Version and configuration management concepts
- Unit testing concepts and practice
- Basics knowledge of Scrum

Literature

1	Mary & Tom Poppendiek: Lean Software Development Kent Beck: eXtreme Programming Explained 2nd Ed.
2	Ken Schwaber et al, Agile Software Development with Scrum, Prentice Hall, 2002
3	Alistair Cockburn: Agile Software Development
4	Robert C. Martin: Agile Software Development
5	Tom Mens: Software Evolution
6	Doug Schmidt et.al.: Pattern-oriented Software Architecture, Vol. 2 Frank Buschmann et al: Pattern-oriented Software Architecture, Vol. 4

7	Len Bass, Paul Clements, Rick Kazman: Software Architecture in Practice 2nd Ed.
8	Gernot Starke: Effektive Software Architekturen 2. Auflage
9	Lehmann "Laws of Software Evolution Revisited"
10	Martin Fowler et al, Refactoring Joshua Kerievsky, Refactoring to Patterns
11	Michael Feathers, Working Effectively with Legacy Code
12	Andreas Zeller: Why Programs Fail ISBN 1558608664

Assessment

Certification requirements for final examinations (conditions for attestation)

Assignments are completed in time and passed. Assignments can be oral presentations and written.

Written module examination

Duration of exam :	120 minutes
Permissible aids:	Any paper documentation