Module Description, available in: EN

Software Engineering and Architectures

General Information

Number of ECTS Credits

3

Module code

TSM_SoftwEng

Valid for academic year

2024-25

Last modification

2023-11-13

Coordinator of the module

Sandy Ingram (HES-SO, sandy.ingram@hefr.ch)

Explanations regarding the language definitions for each location:

- Instruction is given in the language defined below for each location/each time the module is held.
- Documentation is available in the languages defined below. Where documents are in several languages, the percentage distribution is shown (100% = all the documentation).
- The examination is available 100% in the languages shown for each location/each time it is held.

	Lausanne			Lugano	Zurich		
Instruction	**X** E 100%				**X** E 100%		
Documentation	**X** E 100%				**X** E 100%		
Examination	**X** E 100%				**X** E 100%		

Module Category

TSM Technical scientific module

Lessons

2 lecture periods and 1 tutorial period per week

Entry level competences

Prerequisites, previous knowledge

- 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
- Basic knowledge of Scrum

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.

Aims, content, methods

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 (20%)

- Agile Development
  - value creation and innovation
  - risk and complexity in software development
  - agile competence pyramid
- Principles and Practices
  - effective communication among stakeholders
  - project retrospectives, feedback techniques
  - ongoing requirements, solicitation and management
- Modern Software Engineering Methodologies
  - benefits/difficulties and comparison of different approaches (i.e. plan-driven, hybrid and agile)
  - implications for project management
  - incremental planning
- Research in Agile Development

Software Architectures (50%)

- 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
- Architectural patterns
  - for distributed architectures
  - architecture patterns vs design patterns
  - pattern styles
- Selection, creation, and evaluation of software architectures
  - quality attributes
  - architecture analysis methods
- Research in Software Architecture

Software Evolution (30%)

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

Teaching and learning methods

Self-study, homework assignments

Literature

1. Mary & Tom Poppendiek: Lean Software Development
2. Kent Beck: eXtreme Programming Explained 2nd Ed.
3. Ken Schwaber et al, Agile Software Development with Scrum, Prentice Hall, 2002
4. Alistair Cockburn: Agile Software Development
5. Robert C. Martin: Agile Software Development
6. Tom Mens: Software Evolution
7. Doug Schmidt et.al.: Pattern-oriented Software Architecture, Vol. 2
   Frank Buschmann et al: Pattern-oriented Software Architecture, Vol. 4
8. Len Bass, Paul Clements, Rick Kazman: Software Architecture in Practice 2nd Ed.
9. Gernot Starke: Effektive Software Architekturen 2. Auflage
10. Lehmann "Laws of Software Evolution Revisited"
11. Martin Fowler et al, Refactoring
    Joshua Kerievsky, Refactoring to Patterns
12. Michael Feathers, Working Effectively with Legacy Code
13. Andreas Zeller: Why Programs Fail ISBN 1558608664

Assessment

Certification requirements

Module uses certification requirements

Certification requirements for final examinations (conditions for attestation)

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

Basic principle for exams

As a rule, all the standard final exams for modules and also all resit exams are to be in written form

Standard final exam for a module and written resit exam

Kind of exam

written

Duration of exam

120 minutes

Permissible aids

Aids permitted as specified below:

Permissible electronic aids

No electronic aids permitted

Other permissible aids

A paper-based personal summary with a length not exceeding 10 pages A4 (recto-verso) ou 20 pages recto. No software documentation will be allowed.

Special case: Resit exam as oral exam

Kind of exam

oral

Duration of exam

30 minutes

Permissible aids

No aids permitted