Name of the course: Software Technology	**Total credits: 2+2+1=5**

IPM-18AUTSTEG

Type: Obligatory

Total hours per semester
- lecture: 26
- practice: 26
- consultation: 13

Type of testing: Exam
Other: projects in teamwork, test

Semester: 1st

Description

The course gives a broad overview of the processes and methodologies of software development and its execution. We cover all phases of development from requirements to maintenance and quality assurance with emphasis on architectural design.

Main topics: Software Lifecycle and Software Development Process, Software Development Models (from Waterfall to Agile), role of UX, DevOps tool-chain, ways of code verification, OOP, Modelling and Design, SOLID, UML and xtUML, Architectural Improvements, Creational, Structural, Behavioral and Concurrency Design Patterns, Architectural Patterns

Literature

Compulsory

- Ian Sommerville: *Software Engineering.*Pearson Education Limited 2007. ISBN 978-0137035151
- E.Gamma, R.Helm, R.Johnson, J. Vlissides: *Design Patterns: Elements of reusable Object-Oriented Softare*, ISBN 0201633612, Pearson Education Inc., Adison Wesley Professional, 1995.

Recommended

- Martin Fowler: *UML Distilled* (3rd ed.), Addison-Wesley, 2003. ISBN 978-0137035151
- Hassan Gomaa: *Software Modeling and Design: UML, Uses Cases, Patterns and Software Architectures*, Cambridge, 2011. ISBN 978-0521764148
- Steve McConnell: *Software Project Survival Guide*, Microsoft Press, 1997. ISBN 978-1572316218oftware
- Russ Miles: *Learning UML 2.0*, O'Reilly, 2006. ISBN 978-1572316218
- Ron Patton: *Software Testing*, Sams, 2005. ISBN 978-0672327988

Competencies

Knowledge

- Possession of complex and up-to-date knowledge in software technology, regarding the design, implementation, operation and maintenance of software, in the following areas: software architectures and design patterns; model-driven software development; UML and its application in object-oriented and component based design; embedded and real-time systems; reliability and validation of software; testing techniques.
- Knowledge on the methods of architecture description and design.
- Detailed and expert-level knowledge of the technical terms and expressions of computer science in English.

Competencies

- Expertise in the application of the concepts and methods of software technology in modeling of complex software and architecture design.
- Ability to formalize complex technical problems, to analyze theoretical and practical background, and to provide adequate solutions.
- Expertise in design, development, operation and management tasks in the domain of complex software systems.
- Skills for cooperation and team work, and ability to take leading role.
- Ability for written and oral communication in English, using the technical terms and expressions of computer science. Ability to argue, to prepare reports, to read, understand and exploit scientific and technical material (e.g. books and papers).
- Expertise in utilizing sources of technical information, their critical interpretation and evaluation, and the extraction of information relevant to the solution of a specific problem.

Attitude

- Attends professional, technological development related to their qualification.
- Commitment to critical feedback and self-assessment.
- Commitment to lifelong learning and receptivity to new IT competencies.
- Adopts and coordinates the ethical principles of work, organizational culture and research.
- Commitment to quality standards and its IT tools.
- Open to initiate collaboration with IT and other specialists.

Autonomy and responsibility

- Takes responsibility for his professional decisions taken during his professional activities.
- Takes responsibility for observing and enforcing deadlines.
- Takes responsibility for own and fellow workers' work.
- In the case of operational critical IT systems, he/she can be assigned responsibility for development and operation, according to his/her professional competencies.