Study programme(s): Applied Mathematics – Data Science

Level: master studies

Course title: Software engineering

Lecturers: Zoran D. Budimac

Status: elective

ECTS: 5

Requirements: none

Learning objectives

Overview of elementary and advanced phases and techniques of software development. Preparation of students for teamwork in characteristic phases of software development: requirements, analysis, design, implementation, elements of management and quality control.

Learning outcomes

Minimal: Students should be able to apply the obtained knowledge, and be able to work as a team member on the development and delivery of software products of high quality.

Optimal: Students should have good knowledge, ability for critical analysis and application of knowledge in the field, ability to work both individually and as a team member on the development and delivery of high quality software products, as well as the ability to analyze their quality level.

Syllabus

Theoretical instruction

Basic notions and definitions. Software quality criteria. Models of software development process and basic concepts of the development description. Possible views on the software development process: functional, data oriented, rule oriented, state oriented, scenario based. Structure and object-oriented analysis and design. Formal specification. Principles and methods of implementation. Reverse engineering. Standardization of a software development process.

Practical instruction

Analysis and practical improvement of requirements specification. Training in methods of software cost estimation. Training in object-oriented analysis. Training in description of software product by methods of formal specification.

Practical work on system and functional testing. Principles of software metrics and practicing of methods of software quality measurement.

Literature

1. Eric J. Braude, Michael E. Bernstein, Software Engineering: Modern Approaches, John Wiley and sons, 2010

2. R. Pressman: Software Engineering, A Practitioner's Approach, 7th edition, McGraw-Hill, 2009

3. I. Sommerville: Software Engineering, 9th Edition, Addison-Wesley, 2010

4. G. Booch, I.Jacobson, J. Rumbaugh: The Unified Modeling Language User Guide. Addison-Wesley, 2005

Weekly teaching load				**Other:**
Lectures: 2	Exercises: 2	Other forms of teaching:	Student research:	

Teaching methodology

Classical methodology is applied in lectures including the use of the video-beam. During exercises, case studies are analyzed in-depth. Some aspects and principles are practically covered by software tools. Furthermore, students study some of the covered topics and report on their findings in written papers in an individual and more thorough manner.

Grading (maximum number of points 100)

Pre-exam obligations	**points**	**Final exam**	**Points**
Four colloquia	**10, 10, 10, 10**	Written exam	
Six practical assignments	**20**	Oral exam	**40**