Software engineering - course description

General information

Course name	Software engineering
Course ID	11.3-WE-INFP-SoftEng-Er
Faculty	Faculty of Computer Science, Electrical Engineering and Automatics
Field of study	Computer Science
Education profile	academic
Level of studies	First-cycle Erasmus programme
Beginning semester	winter term 2019/2020

Course information

Semester	4
ECTS credits to win	6
Course type	obligatory
Teaching language	english
Author of syllabus	• dr inż. Tomasz Gratkowski • mgr inż. Marcin Andrzejewski • dr inż. Michał Doligalski

Classes forms

The class form	Hours per semester (full-time)	Hours per week (full-time)	Hours per semester (part-time)	Hours per week (part-time)	Form of assignment
Lecture	30	2	-	-	Credit with grade
Project	30	2	-	-	Credit with grade

Aim of the course

- Familiarize students with methods of design, analysis and methods of testing programs.
- Teach students the fundamental skills in specification requirements, planning, documentation of IT projects.
- Familiarize students with with tools for object-oriented design and project management.

Prerequisites

Algorithms and data structures, Principles of programming, Object-oriented programming

Scope

Introduction to software engineering. Why engineering software is different? Software lifespan and maintenance. Lifecycle models with specified project phases. Information systems. System and software design. Models for information systems. Software process. Requirements analysis and specification. Guidelines and forms for specification. Design. Purpose of design. Fundamental design concepts. Design strategies. Design quality metrics. Reliability and system security. Implementation. Review of structural programming. Error handling and defensive programming. Aids to maintainability. Coding for performance. Testing. Reasons for testing. Black box and structural testing. Testing strategies. Tools for testing Computer Aided Software Engineering tools. Upper and Lower CASE, CASE workbenches.

Teaching methods

Lecture, project

Learning outcomes and methods of theirs verification

Outcome description	Outcome symbols	Methods of verification	The class form
Understands software distribution and maintenance problems, can work and communicate in a programming team		• a test with score scale	• Lecture • Project
Can develop a project plan, documentation of requirements, requirement specification as well as functional and program specification, can also evaluate the quality of a project using appropriate tools		• a project • an observation and evaluation of the student's practical skills	• Project
Can define and characterize basic software production cycles		• a project • an observation and evaluation of the student's practical skills	• Lecture • Project

Assignment conditions

Lecture - obtaining a positive grade in written exam.

Project - a condition of pass is to obtain positive marks from all project tasks and preparation written report of project.

Calculation of the final grade: = lecture 50% + project 50%.

Recommended reading

1. Ian Sommerville: Software Engineering (10th Edition), Pearson, 2015
2. Joel Spolsky: The Best Software Writing I: Selected and Introduced, Apress, 2005
3. Len Bass, Paul Clements, Rick Kazman: Software Architecture in Practice, Second Edition, Addison-Wesley Professional, 2003
4. Steve McConnell: Code Complete: A Practical Handbook of Software Construction, Second Edition 2nd Edition, Microsoft Press, 2004
5. Martin Fowler, Kent Beck, John Brant, William Opdyke, and Don Roberts: Refactoring - Improving the Design of Existing Code, Addison-Wesley Professional, 1999
6. Eric Freeman, Bert Bates, Kathy Sierra, Elisabeth Robson: Head First Design Patterns: A Brain-Friendly Guide 1st Edition, O'Reilly Media, 2004

Further reading

Notes

Modified by prof. dr hab. inż. Andrzej Obuchowicz (last modification: 27-10-2019 10:48)