



COURSE DESCRIPTION CARD - SYLLABUS

Course name

Software Development Studio 1 [S2Inf1E-IO>SDS1]

Course

Field of study

Computing

Year/Semester

1/1

Area of study (specialization)

Software Engineering

Profile of study

general academic

Level of study

second-cycle

Course offered in

English

Form of study

full-time

Requirements

compulsory

Number of hours

Lecture

0

Laboratory classes

0

Other

0

Tutorials

0

Projects/seminars

75

Number of credit points

6,00

Coordinators

dr hab. inż. Mirosław Ochodek prof. PP
miroslaw.ochodek@put.poznan.pl

dr inż. Sylwia Kopczyńska
sylwia.kopczynska@put.poznan.pl

Lecturers

Prerequisites

Student starting this course shall have a basic knowledge concerning project management, requirements engineering, and programming. Also, they should be able to acquire information from the recommended sources and be willing to cooperate within a project team.

Course objective

1. Provide to the students the foundations of project management and requirements engineering (and illustrate them with real-life cases) that are necessary to perform the management and analytical roles in software development projects, 2. Development of the skills related to the software product development (especially, the skills related to project management, requirements engineering, and software architecture) by participation in a capstone project. The goal of the project is to solve a problem of a real customer. The main focus is on developing skills related to the preparation of business case, project initiation, elicitation and documentation of requirements, and software architectures. 3. Development of communication and team-work skills.

Course-related learning outcomes

Knowledge:

1. has organized and well-formed theoretical general knowledge regarding agile software development methods (e.g., the agile manifesto and the most popular agile methods).
2. has advanced and detailed knowledge regarding scrum.
3. has advanced and detailed knowledge regarding planning software-product development (e.g., product-backlog management).
4. has advanced and detailed knowledge about software-development process proposed by scrum.

Skills:

1. is able to acquire knowledge necessary to conduct a software-development project (technical and domain knowledge).
2. is able to use task-management tools (e.g., jira) and customize them depending on project needs.
3. is able to integrate technical and domain knowledge to specify requirements for a software product.
4. is able to evaluate usefulness of technical solutions for the development of a software product.

Social competences:

1. is aware of directions and intensity of modern software-development methodologies.
2. understands the necessity of continuous development of methods and ict technologies and the necessity of adapting them to the context of a software-development project.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Learning outcomes presented above are verified as follows:

Formative assessment:

- based on the regular assessment of the provided information regarding the activities performed within the project tasks
- based on the regular assessment of the current status of software development project tasks

Summative assessment:

Performed based on four criteria (the average percentage points):

- active participation in course classes (average grade is taken => 0-100%);
- knowledge test (mainly prepared based on the scrum.org certification tests) (average grade is taken => 0-100%);
- application of good practices (their selection depends on a project context) (0-100%);
- the quality of delivered products (min. business case and software requirements specification; other products selected based on the project context) (0-100%).

The final grade is determined using the following scale:

- (90%, 100%] - 5.0
- (80%, 90%] - 4.5
- (70%, 80%] - 4.0
- (60%, 70%] - 3.5
- (50%, 60%] - 3.0
- (0%, 50%] - 2.0

Programme content

Within the Software development studio course (part I and II), the students take part in a capstone software project aiming at solving a real-life problem defined by an external customer. The course covers the initial stages of a software development project and is continued in the following semester within the Software Development Studio 2 course.

The projects are run according to Scrum. A student can fulfill the Scrum Master, Product Owner or architect roles, while students of the first cycle studies in computer science play the roles of software developers.

The following aspects are covered within the course:

- agile software development methods with a special focus on Scrum (roles, artifacts, events);
- planning and managing product development (Product Backlog, the Kano model);
- applying Design Thinking to software development.

Course topics

1. Organizational meeting – presentation of the course rules and structure.
2. Thematic workshops conducted with the entire class group.
3. Regular mentor meetings (weekly) – presentation of project progress, discussion of encountered problems and mitigation strategies, risk analysis, and planning of next steps.
4. Team building – attracting other students to the project topic, recruitment, and applying team-integration practices.
5. Project supervision, including:
 - developing the business case,
 - eliciting, analyzing, and documenting requirements,
 - designing the initial software architecture,
 - planning and conducting sprints,
 - ensuring the quality of Scrum events (Daily Stand-up, Sprint Planning, Sprint Review, Sprint Retrospective) and meetings with the client,
 - monitoring project progress and the quality of produced artifacts.

Teaching methods

A capstone project teaching method is the main teaching method of choice, which is augmented with the use of multimedia presentations and case studies during the tutorial part.

More information about the teaching method used in the course can be found in the paper:

Kopczyńska, Sylwia, Jerzy Nawrocki, and Mirosław Ochodek. Software development studio: bringing industrial environment to a classroom. Proceedings of the First International Workshop on Software Engineering Education Based on Real-World Experiences. IEEE Press, 2012.

Bibliography

Basic

1. K. Schwaber, J. Sutherland, The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game, <http://www.scrumguides.org>, (available online), 2017

Additional

1. Nawrocki, Jerzy, et al. Agile requirements engineering: A research perspective. International Conference on Current Trends in Theory and Practice of Informatics. Springer, Cham, 2014.

Breakdown of average student's workload

	Hours	ECTS
Total workload	150	6,00
Classes requiring direct contact with the teacher	75	3,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	75	3,00