STUDY MODULE DESCRIPTION FORM

Name of the module/subject **Software Development Studio 1**	Code **1010512311010513906**

Field of study **Computing**	Profile of study (general academic, practical) **general academic**	Year /Semester **1 / 1**
Elective path/specialty **Software Engineering**	Subject offered in: **English**	Course (compulsory, elective) **obligatory**

Cycle of study: **Second-cycle studies**	Form of study (full-time,part-time) **full-time**

No. of hours Lecture: **-** Classes: **-** Laboratory: **-** Project/seminars: **60**	No. of credits **6**

Status of the course in the study program (Basic, major, other) **major**	(university-wide, from another field) **from field**

Education areas and fields of science and art **technical sciences**	ECTS distribution (number and %**)** **6 100%**

Responsible for subject / lecturer:

Mirosław Ochodek, PhD
email: Miroslaw.Ochodek@put.poznan.pl
tel. 61 6652944
Institute of Computing Science
Piotrowo 2 Str., 60-965 Poznan

Prerequisites in terms of knowledge, skills and social competencies:

1	**Knowledge**	Learning objectives of the first cycle studies defined in the resolution of the PUT Academic Senate, especially K_W1-2, K_W4, K_W6-15 that are verified in the admission process to the second cycle studies ? the learning objectives are available at the website of the faculty www.fc.put.poznan.pl
2	**Skills**	Learning objectives of the first cycle studies defined in the resolution of the PUT Academic Senate, especially K_U1-2, K_U4, K_U7-8, K_U14-20, K_U22-23, K_U26 that are verified in the admission process to the second cycle studies ? the learning objectives are available at the website of the faculty www.fc.put.poznan.pl
3	**Social competencies**	Learning objectives of the first cycle studies defined in the resolution of the PUT Academic Senate, especially K_K1-9 that are verified in the admission process to the second cycle studies ? the learning objectives are available at the website of the faculty www.fc.put.poznan.pl In addition, in respect to the social skills the student should show attitudes as honesty, responsibility, perseverance, curiosity, creativity, manners, and respect for other people.

Assumptions and objectives of the course:

1. Provide knowledge, supported with the presentation of real cases, regarding software engineering related to software project management, requirements engineering, software architecture, which is necessary to fulfill the roles of: project manager, analyst, architect,

2. Develop students? skills in solving problems related to software project management, requirements engineering, and software architecture by involving students in a software project developed for a real customer. Within the course, the main focus is on developing student?s skills related to preparation of project assumptions, initiation of a project, elicitation and analysis of requirements, development of software requirements specification, and design and documentation of software architecture.

3. Develop students? teamwork skills.

Study outcomes and reference to the educational results for a field of study

Knowledge:

http://www.put.poznan.pl/

1. has well-established theoretical knowledge of software engineering regarding project management, functional and non-functional requirements elicitation, software architecture, etc. - [K_W4]

2. has detailed theoretical knowledge related to the selected areas of computer science including: software project management according to PRINCE2, eXtreme Programming, and XPrince; functional requirements specification with the use of use cases; organizing and moderating architecture analysis workshops, and designing acceptance tests for software systems. - [K_W5]

3. has basic knowledge regarding life-cycle of software systems, especially regarding the life-cycle of the software development stages. - [K_W7]

4. knows the fundamental methods, techniques and tools employed to solve complex engineering tasks related to management and planning of software projects, requirements elicitation, GUI design, software and process measurement, and acceptance testing. - [K_W8]

Skills:

1. is able to acquire, combine, interpret and evaluate information from literature, databases and other information sources (in mother tongue and English); draw conclusions, and formulate opinions based on it. - [K_U1]

2. is able to: participate and moderate software project meeting (in mother tongue and English) regarding various aspects of software development, present assumptions, constraints, current status and the results of a software project, negotiate with a client, conduct requirements elicitation workshops, use software tools supporting communication among team members (e.g., Redmine). - [K_U2]

3. is able to plan and arrange self-education process especially regarding new methods, techniques, and technologies used in software development. - [K_U5]

4. is able to understand the content of books and papers in English, can play the role of a moderator during a project meeting in English, can present the assumptions, the results of a project and negotiate with a client in English. - [K_U6]

5. is able to use Information and Communication Technologies that are commonly employed in IT projects when participating in a software project. - [K_U7]

6. is able to combine knowledge from different areas of computer science (and if necessary from other scientific disciplines) to formulate and solve engineering tasks; and use system approach that also incorporates nontechnical aspects - [K_U10]

7. is able to conduct risk analysis of an IT project and manage the risk regarding the software project he/she participate in especially in the early stages of the project. - [K_U11]

8. is able to assess usefulness and possibility of employing new developments (methods and tools) and new IT products. - [K_U13]

9. is able to correctly apply at least one software effort estimation method - [K_U15]

10. is able to develop an object-oriented model of a simple software system (e.g., in UML notation) - [K_U17]

11. is able to assess software architecture from the perspective of non-functional requirements - [K_U18]

12. is able to design acceptance tests regarding functional and non-functional aspects of a software product. - [K_U20]

13. is able to propose enhancements (improvements) to existing technical solutions. - [K_U21]

14. is able to prepare use-case-based functional requirements specifications. - [K_U22]

15. is able to formulate non-functional requirements for selected quality characteristics (categories). - [K_U23]

Social competencies:

1. is able to collaborate and cooperate in a team performing different roles including the roles of project manager, analyst, architect. - [K_K5]

2. is able to correctly assign priorities to own tasks and tasks performed by others. - [K_K6]

3. is able to think and act in an entrepreneurial way. - [K_K8]

Assessment methods of study outcomes

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,
- based on the regular assessment of the conduction and organization of software project meetings (meetings with team members, the representatives of client, user, supplier);
- based on the regular assessment of the project documentation.

Summative assessment:

- based on the assessment of the student?s preparation to the project classes (multiple choice question test) ? K_W4, K_W5, K_W7, K_W8, K_U1, K_U6;
- based on the assessment of the quality of the created project documentation (management and planning documents, requirements specification documents, design documents of software architecture, test specifications, and other technical documentation) ? K_W4, K_W5, K_W7, K_W8, K_U1, K_U5, K_U6, K_U10, K_U13, K_U15, K_17, K_U18, K_U21, K_U22, K_U23;
- based on the assessment of project planning and following a project schedule ? K_U27, K_K6, K_K8;
- based on the assessment of applying good software engineering practices in a software project the student participates in, including the responsibilities assigned to the roles performed by the student ? K_W4, K_W5, K_W7, K_W8, K_U2, K_U5, K_U7, K_U20, K_K2, K_5;
- based on the assessment of the effectiveness of communication among team members based on the task log in a task management tool - K_U2, K_U7;
- based on the assessment of the quality of risk log and the effectiveness of risk mitigation actions ? K_U11;
- based on the assessment of the quality of development team project meetings (including architecture evaluation meetings) and meetings with the representatives of customer, user, and supplier ? K_U2, K_U6, K_U7, K_U18;
- based on the assessment of the completeness and quality of the acceptance tests ? K_U20.

Course description

During the course a student takes part in a real software project aiming at solving a real problem belonging to a real customer. The course covers the initial stages of a software development project which is continued in the following semester within the Software Development Studio 2 course.

The projects are run according to the guidelines of XPrince project management methodology which is based on three well-known methodologies: PRINCE2, Rational Unified Process, and eXtreme Programming. Each student can play one or more of the following project roles: project manager, analyst, and architect. The students of first cycle in computer science play roles of software developers.

The following project stages are covered within the scope of the course: starting up a project, initiating a project, and software architecture elaboration.

The following aspects are covered within the course:

- Team structure (steering committee, responsibilities of the roles of project manager, analyst, architect);
- Software project life-cycle (XPrince, PRINCE2, eXtreme Programming ? initial stages of a project and the tasks that should be completed in each stage);
- Project management, controlling a stage (evaluating business value of a project, planning a project, a stage, tasks delegation in a team);
- Elicitation and analysis of requirements (use cases, non-functional requirements, GUI workshops, requirements elicitation workshops);
- Estimation of software size and software development effort;
- Quality assurance (acceptance tests regarding functional and non-functional aspects of a system, measurement program);
- Software architecture (design, decisions vs. non-functional requirements, architecture analysis);
- Selection and development of project infrastructure, environment (infrastructure for task management, planning, communication);
- Team building (recruitment to IT projects, good practices supporting effective communication in a team);
- Configuration management (planning, identification, control of configuration items, development of the infrastructure);
- Reflection workshops;
- Risk management (risk identification at early stages of software project, analysis and mitigation).

Learning methods:

1. multimedia presentation, multimedia showcase, discussion, teamwork, workshop, case studies, tutorial.

Basic bibliography:

1. Office of Government Commerce, An Introduction to PRINCE2: Managing and Directing Successful Projects, The Stationary Office, ISBN-13: 9780113311880, 2009

2. Writing Effective Use Cases, A. Cockburn, Addison-Wesley, Boston, 2001

http://www.put.poznan.pl/

Additional bibliography:

1. Software Requirements 2nd edition, K. E. Wiegers, Microsoft Press, Redmond, WA, USA, 2003
2. 7 Habits of Highly Effective People, S. Covey, Free Press, London, 2004
3. Len Bass, Paul Clements, Rick Kazman, Software Architecture in Practice (2nd Edition), ISBN-13: 978-0321154958, Addison-Wesley Professional, 2003

Result of average student's workload

Activity	Time (working hours)
1. participating in project classes: 15 x 4 hours,	60
2. preparing to project classes (15 x ~0.5 hour),	7
3. consulting issues related to the subject of the course; especially related to classes and projects,	5
4. participating in project team meetings (13 x 1 hour),	13
5. participating in meetings with steering committee (10 x 2 hours),	20
6. project management (15 x 1 hour),	15
7. development of project documentation (20x 1 hour),	20
8. studying literature / learning aids (10 pages = 1 hour), 100 pages.	10

Student’s workload

Source of workload	hours	ECTS
Total workload	150	6
Contact hours	98	4
Practical activities	140	6