Course Syllabus

I. General Information

Course name	Software engineering
Programme	Computer science
Level of studies (BA, BSc, MA, MSc, long-cycle MA)	BA
Form of studies (full-time, part-time)	Full-time
Discipline	Computer science
Language of instruction	English

Course coordinator	Rafał Lizut

Type of class *(use only the types mentioned below)*	Number of teaching hours	Semester	ECTS Points
lecture	30	5	5
tutorial			
classes			
laboratory classes	30	5	
workshops			
seminar			
introductory seminar			
foreign language classes			
practical placement			
field work			
diploma laboratory			
translation classes			
study visit			

Course pre-requisites	Knowledge of structural and object-oriented programming

II. Course Objectives

Raising the level of knowledge of students in the field of software engineering
Presentation and detailed discussion of all aspects of software development from the initial phase of the specification up to its maintenance after the date of commencement of use
Familiarize students with their ability to work in accordance with structural, object and agile methodologies

III. Course learning outcomes with reference to programme learning outcomes

Symbol	Description of course learning outcome	Reference to programme learning outcome
	KNOWLEDGE	
W_01	The student knows what is software engineering, the process of software development, project management	K_W04, K_W06
W_02	The student knows how software requirements should be set, how the requirements engineering process looks like, system modeling, software prototyping, verification, testing and acceptance of approved software	K_W04, K_W06
W_03	The student knows what are the methods of personnel management, quality management, software estimation, software upgrading	K_W04, K_W06
	SKILLS	
U_01	The student constructs non-functional requirements and prepare software specifications	K_U02, K_U04, K_U13, K_U14, K_U17, K_U23, K_U29, K_U30
U_02	Student uses diagrams describing the structure and behavior of the program	K_U02, K_U04, K_U13, K_U14, K_U23, K_U30
U_03	The student uses the UML language to the basic level	K_U02, K_U04, K_U13, K_U14, K_U23, K_U30
U_04	Student develops a project plan for software development	K_U02, K_U04, K_U13, K_U14, K_U17, K_U23, K_U29, K_U30
U_05	Student controls and manages versions of the created software and adhere to the rules of existing programmers while working in a team	K_U02, K_U04, K_U13, K_U17, K_U23, K_U30
	SOCIAL COMPETENCIES	
K_01	Student recognizes to the complexity of problems with which he may meet in life	K_K01, K_K02, K_K04, K_K05
K_02	Student skillfully solves software engineering problems using known methods and objectively evaluates obtained results	K_K01, K_K02, K_K04, K_K05
K_03	Student is able to work both individually and as a team, properly planning his and the team's work in the context of the set goals	K_K01, K_K02, K_K04, K_K05

IV. Course Content

1 Introduction
2 Software development processes
3 Requirements engineering
4 Structural methods
5 Object-oriented methods
6 Basics of UML 6 Code quality, code inspections
7 Testing

8 User documentation 9 Maintenance 10 Critical systems 11 Formal methods 12 Design patterns

V. Didactic methods used and forms of assessment of learning outcomes

Symbol	Didactic methods *(choose from the list)*	Forms of assessment *(choose from the list)*	Documentation type *(choose from the list)*
		KNOWLEDGE	
W_01	Conventional lecture/Problem lecture	Exam	Evaluated test / written test
W_02	Conventional lecture/Problem lecture	Exam	Evaluated test / written test
W_03	Conventional lecture/Problem lecture	Exam	Evaluated test / written test
		SKILLS	
U_01	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
U_02	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
U_03	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
U_04	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
U_05	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
		SOCIAL COMPETENCIES	
K_01	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
K_02	Project-based learning design thinking	Preparation / implementation of the project	Project rating card
K_03	Project-based learning design thinking	Preparation / implementation of the project	Project rating card

VI. Grading criteria, weighting factors

90 – 100% - very good (5.0),
80 – 89% - good plus (4.5),
70 – 79% - good (4.0),

60 – 69% - satisfactory plus (3.5),
50 – 59% - satisfactory (3.0),
Less than 50% - unsatisfactory (2.0).

VII. Student workload

Form of activity	Number of hours
Number of contact hours (with the teacher)	***80***
Number of hours of individual student work	***60***

VIII. Literature

Basic literature
1. Sommerville, Ian (2007) [1982]. "1.1.2 What is software engineering?". Software Engineering (8th ed.). Harlow, England: Pearson Education. p. 7. ISBN 0-321-31379-8. 2.Peter, Naur; Randell, Brian (7–11 October 1968). Software Engineering: Report of a conference sponsored by the NATO Science Committee (PDF). Garmisch, Germany: 3. Scientific Affairs Division, NATO. Retrieved 2008-12-26.2018 International Conference on Software Engineering celebrating its 40th anniversary, and 50 years of Software engineering. "ICSE 2018 - Plenary Sessions - Margaret Hamilton". Retrieved 9 Aug 2018. 4."Software Engineering Body of Knowledge (SWEBOK Version 3), 2014" (pdf). www.swebok.org. IEEE Computer Society. Retrieved 24 May2016. 5. Abran, Alain, ed. (2005) [2004]. "Chapter 1: Introduction to the Guide". Guide to the Software Engineering Body of Knowledge. Los Alamitos: IEEE Computer Society. ISBN 0-7695-2330-7. Retrieved 2010-09-13. 6. http://staruml.sourceforge.net/en/ StarUML - The Open Source UML/MDA Platform 7. http://cnx.org/content/m15092/latest/ StarUML Tutorial 8. http://www.microtool.de/objectif/en/index.asp objectiF - Tool for Model-Driven Software Development with UML 9. http://www.microtool.de/mT/pdf/objectiF/01/Tutorials/JavaTutorial.pdf Developing Java Applications with UML 10. Abran, Alain; Moore, James W.; Bourque, Pierre; Dupuis, Robert; Tripp, Leonard L. (2004). Guide to the Software Engineering Body of Knowledge. IEEE.ISBN 0-7695-2330-7. 11. Sommerville, Ian (2008). Software Engineering (7 ed.). Pearson Education. ISBN 978-81-7758-530-8. Retrieved 10 January 2013.
Additional literature
1.G. Mathew, A. Agrawal, and T. Menzies, “A Method for Finding Trends in Software Research,” 2018; https://arxiv.org/pdf/1608.08100.pdf. 2.K.-Y. Cai and D. Card, “An Analysis of Research Topics in Software Engineering—2006,” J. Systems and Software, vol. 81, no. 6, 2008, pp. 1051–1058. 3. V. Garousi and G. Ruhe, “A Bibliometric/Geographic Assessment of 40 Years of Software Engineering Research (1969–2009),” Int’l J. Software Eng. and Knowledge Eng., vol. 23, no. 9, 2013, pp. 1343–1366. 4. S. Datta, S. Sarkar, and A. Sajeev, “How Long Will This Live? Discovering the Lifespans of Software Engineering Ideas,” IEEE Trans. Big Data, vol. 2, no. 2, 2016, pp. 124–137.