KARTA PRZEDMIOTU

I. Dane podstawowe

Nazwa przedmiotu	Software engineering
Nazwa przedmiotu w języku angielskim	Software engineering
Kierunek studiów	Informatyka
Poziom studiów (I, II, jednolite magisterskie)	I
Forma studiów (stacjonarne, niestacjonarne)	Stacjonarne
Dyscyplina	Informatyka
Język wykładowy	Angielski

Koordynator przedmiotu/osoba odpowiedzialna	Marcin Płonkowski

Forma zajęć *(katalog zamknięty ze słownika)*	Liczba godzin	semestr	Punkty ECTS
wykład	30	5	5
konwersatorium			
ćwiczenia	30	5	
laboratorium			
warsztaty			
seminarium			
proseminarium			
lektorat			
praktyki			
zajęcia terenowe			
pracownia dyplomowa			
translatorium			
wizyta studyjna			

Wymagania wstępne	Knowledge of structural and object-oriented programming

II. Cele kształcenia dla przedmiotu

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. Efekty uczenia się dla przedmiotu wraz z odniesieniem do efektów kierunkowych

Symbol	Opis efektu przedmiotowego	Odniesienie do efektu kierunkowego
WIEDZA		
W_01	The student knows what is software engineering, the process of software development, project management	K_W04, K_W06, K_W08
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_W06, K_W08
UMIEJĘTNOŚCI		
U_01	The student can construct non-functional requirements and prepare software specifications	K_U02, K_U04, K_U13, K_U14, K_U23
U_02	Student is able to use diagrams describing the structure and behavior of the program	K_U02, K_U04, K_U13, K_U14, K_U23
U_03	The student is able to use the UML language to the basic level	K_U02, K_U04, K_U13, K_U14, K_U23
U_04	Student is able to develop a project plan for software development	K_U02, K_U04, K_U14, K_U23
U_05	Student is able to control and manage 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_U14, K_U23, K_U29, K_U30
KOMPETENCJE SPOŁECZNE		
K_01	Student is open to the complexity of problems with which he may meet in life	K_K01, K_K06
K_02	Student skillfully solves software engineering problems using known methods and objectively evaluates obtained results	K_K01, K_K03
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 goa	K_K02, K_K04, K_K07

IV. Opis przedmiotu/ treści programowe

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. Metody realizacji i weryfikacji efektów uczenia się

Symbol efektu	Metody dydaktyczne *(lista wyboru)*	Metody weryfikacji *(lista wyboru)*	Sposoby dokumentacji *(lista wyboru)*
WIEDZA			
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
UMIEJĘTNOŚCI			
U_01	Practical classes	Test / Written test	Evaluated test / written test
U_02	Practical classes	Test / Written test	Evaluated test / written test
U_03	Practical classes	Test / Written test	Evaluated test / written test
U_04	Practical classes	Test / Written test	Evaluated test / written test
U_05	Practical classes	Test / Written test	Evaluated test / written test
KOMPETENCJE SPOŁECZNE			
K_01	Practical classes	Test / Written test	Evaluated test / written test
K_02	Practical classes	Test / Written test	Evaluated test / written test
K_03	Practical classes	Test / Written test	Evaluated test / written test

VI. Kryteria oceny, wagi

2 colloquies in the semester
80 % - written answers to test tasks and oral answers in case of doubt,
20% - the grade obtained from the classes.

A grading scale is given below:
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. Obciążenie pracą studenta

Forma aktywności studenta	Liczba godzin
Liczba godzin kontaktowych z nauczycielem	**80**
Liczba godzin indywidualnej pracy studenta	**60**

VIII. Literatura

Literatura podstawowa
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.
Literatura uzupełniająca
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.