

Subject card

Subject name and code	Software Engineering, PG_00143568						
Field of study	Informatics						
Date of commencement of studies	October 2025		Academic year of realisation of subject		2026/2027		
Education level	Bachelor's studies		Subject group		Obligatory subject group in the field of study Subject group related to scientific research in the field of study		
Mode of study	full-time studies		Mode of delivery		at the university		
Year of study	2		Language of instruction		Polish		
Semester of study	4		ECTS credits		3.0		
Learning profile	academic		Assessment form		credit		
Conducting unit							
Name and surname of lecturer (lecturers)	Subject supervisor		dr Adam Kostulak				
	Teachers						
Lesson types	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	15.0	0.0	15.0	0.0	0.0	30
	E-learning hours included: 0.0						
Learning activity and number of study hours	Learning activity	Participation in didactic classes included in study plan		Participation in consultation hours		Self-study	SUM
	Number of study hours	30		0.0		45.0	75
Subject objectives	The aim of the course is to familiarise students with the basic problems, methods, techniques and tools of quality software production quality. Introducing various software life cycle models, software development methods, basics of documentation, requirements analysis design, software testing. Discussion of requirements analysis management, creating specifications with defined metrics. Topics discussed presented with emphasis on object-oriented approach to design and system modelling using UML.						
Learning outcomes	Course outcome		Subject outcome		Method of verification		
	[INFL3_W04] has structured knowledge in the field of software engineering, software specifications, validation and verification, and tools supporting the software development process		students have knowledge of the basics of documentation production, requirements analysis, design, software testing		[SW4] test/exam - oral or written		
	[INFL3_U06] is able to design, create, run and test programs using dedicated tools and adequate templates		Can manage requirements analysis; Can create a specification with defined metrics;t can create a software design using UML		[SU2] presentation/project/paper/report		
	[INFL3_U04] is able to work in a team of IT specialists, manage his time and make commitments and meet deadlines, communicate using various techniques including dedicated tools		students have basic communication skills in a IT team		[SU2] presentation/project/paper/report		

Subject contents	1. introduction; motivations for systematic software development. 2. software life cycle, phases of the classical cycle, place of analysis and design in the development cycle. 3 Requirements. Categories, elicitation, analysis and specification of requirements. Verification and validation; acceptance testing. 4. modelling; world modelling vs. system modelling, place of the model in analysis and design. 5 Use cases. Diagram, interpretation, structured description, examples. 6. Overview of basic principles and concepts of the object-oriented paradigm: object, classification, aggregation, inheritance, communication. Objectivity vs. human's natural way of perceiving reality. 7. Introduction to the UML methodology: class model, dynamic model. Relationships between models. 8 Objects, classes, attributes, operations. Relationships between classes (associations). Relationships as classes. Roles. Aggregation as a special case of a relationship. Semantics of aggregation. Propagation of operations as an aggregation criterion. Examples. 9 Inheritance: specialisation and generalisation. Hierarchy of inheritance. Redefining properties of objects down the inheritance hierarchy, Obscuring Abstract classes. An object as an instance of a class and all its superclasses. 10. How to create a class model? Example of creating a class model. 11. dynamic model: events, actions, activities. Sequence (interaction) diagrams. Linking to m other models. How to create a sequence diagram. 12 State diagram. States: simple, complex, concurrent. inheritance of states. Transitions between states. Linking to the class model. How to create a diagram of a dynamic model? Example of creating a dynamic model. 13 Moving from model to design. Basic phases of object-oriented design. Division into subsystems and modules, identification of concurrency, Processor allocation, resource allocation, priorities. 14 Moving from design to program.		
Prerequisites and co-requisites			
Assessment methods and criteria	Subject passing criteria	Passing threshold	Percentage of the final grade
		51.0%	50.0%
		51.0%	50.0%
Recommended reading	Basic literature	Eriksson H-E, Penker M.: UML Toolkit. Wiley Computer Publishing, John Wiley & Sons, Inc. 1998	
		Pressman R.S.: Software Engineering. A Practitioners Approach. McGraw-Hill, Inc. 1992.	
	Supplementary literature	Booch G., Rumbaugh J., Jacobson I.: UML podręcznik użytkownika. WNT, 2001	
		Eriksson H-E, Penker M.: UML Toolkit. Wiley Computer Publishing, John Wiley & Sons, Inc. 1998.	
	eResources addresses		
Example issues/ example questions/ tasks being completed	Analyse the requirements for the given case study and develop the corresponding UML/BPMN diagrams.		
Work placement	Not applicable		

Document generated electronically. Does not require a seal or signature.