GDAŃSK UNIVERSITY OF TECHNOLOGY

Subject card

Subject name and code	Software Engineering, PG_00044135		
Field of study	Mathematics		
Date of commencement of studies	October 2025	Academic year of realisation of subject	2025/2026
Education level	second-cycle studies	Subject group	Specialty subject group 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	1	Language of instruction	Polish
Semester of study	2	ECTS credits	4.0
Learning profile	general academic profile	Assessment form	assessment
Conducting unit	Institute of Applied Mathematics -> Faculty of Applied Physics and Mathematics -> Faculties of Gdańsk University of Technology		
Name and surname of lecturer (lecturers)	Subject supervisor	dr hab. Paweł Pilarczyk	
	Teachers	dr hab. Paweł Pilarczyk	

Lesson types	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	30.0	0.0	15.0	15.0	0.0	60
	E-learning hours included: 0.0						
	eNauczanie source address: https://enauczanie.pg.edu.pl/moodle/course/view.php?id=46659 Moodle ID: 46659 Inżynieria oprogramowania 2026 https://enauczanie.pg.edu.pl/moodle/course/view.php?id=46659						

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	60	5.0	35.0	100

Subject objectives	Becoming familiar with basic methods of software engineering and acquiring the ability to apply these methods in practice. This concerns group work on a large scale IT project at all the stages of software production: beginning with requirements engineering, through requirements analysis, software design, implementation, testing, installation, to the stage of software maintenance. Also getting acquainted with basic issues regarding IT project management, such as quality management.

Learning outcomes

Course outcome	Subject outcome	Method of verification
[K7_W03] demonstrates knowledge advanced computation techniques, supporting the work of a mathematician and understand their limitations.	Knows software engineering methods for designing, creating, testing and maintaining software.	[SW1] Assessment of factual knowledge [SW2] Assessment of knowledge contained in presentation [SW3] Assessment of knowledge contained in written work and projects
[K7_K03] works as a team; understands the necessity of systematic work on all projects that are long-term in nature, understands and appreciates the importance of intellectual honesty in one's own activities and the activities of other people; behaves ethically	Describes selected software development processes (such as waterfall, iterative, agile). Can develop software in a team. Is able to conduct requirements solicitation, requirements analysis, and to design software. Can use UML to model selected aspects of software. Knows and understands ethics code of a software engineer.	[SK1] Assessment of group work skills [SK3] Assessment of ability to organize work [SK4] Assessment of communication skills, including language correctness [SK5] Assessment of ability to solve problems that arise in practice
[K7_U10] understands the mathematical foundations of the analysis of algorithms and computational processes, constructs algorithms with good numerical properties, used to solve typical and unusual mathematical problems	Can design and create software that satisfies given requirements.	[SU1] Assessment of task fulfilment [SU4] Assessment of ability to use methods and tools

<table>
<tr><td rowspan="3">Subject contents</td><td colspan="3">Course content – lecture
Introduction to software engineering. Ethics code. Software development processes. Requirements engineering. Structural and object-oriented methods. Using the UML in modelling. Testing software. IT project management, quality management. Software maintenance. Critical systems. Agile Manifesto and agile methods, including Scrum.</td></tr>
<tr><td colspan="3">Course content – laboratory
Applying the software engineering methods to designing specific software, including preparation of UML diagrams and user stories.</td></tr>
<tr><td colspan="3">Course content – project
Practical development of software prototypes complying with specified requirements.</td></tr>
<tr><td>Prerequisites and co-requisites</td><td colspan="3">Ability to write computer programs, e.g. in Python, C, or HTML/JavaScript.</td></tr>
<tr><td rowspan="3">Assessment methods and criteria</td><td>Subject passing criteria</td><td>Passing threshold</td><td>Percentage of the final grade</td></tr>
<tr><td>quizzes (in writing, 10 minutes each)</td><td>60.0%</td><td>50.0%</td></tr>
<tr><td>laboratory assignments, project, presentations</td><td>60.0%</td><td>50.0%</td></tr>
<tr><td rowspan="3">Recommended reading</td><td>Basic literature</td><td colspan="2">Krzysztof Sacha, Inżynieria oprogramowania, PWN 2010.
Mariusz Chrapko, Scrum. O zwinnym zarządzaniu projektami. Wydanie II rozszerzone, Helion 2014.</td></tr>
<tr><td>Supplementary literature</td><td colspan="2">Ian Sommerville, Software Engineering, Pearson, 10th edition, 2015.
Roger S. Pressman, Software Engineering: A Practitioner's Approach, McGraw Hill; 8th edition, 2014.
Russ Miles, Kim Hamilton, Learning UML 2.0: A Pragmatic Introduction to UML, O'Reilly and Associates, 2006.
Kenneth S. Rubin. Essential Scrum: A Practical Guide to the Most Popular Agile Process. Pearson Education, 2013.</td></tr>
<tr><td>eResources addresses</td><td colspan="2"></td></tr>
<tr><td>Example issues/ example questions/ tasks being completed</td><td colspan="3">Requirements engineering.
Software development processes.
Verification and validation.
Agile methods and the Manifesto.
Ethics code of a software engineer.
The INVEST features of user stories.</td></tr>
<tr><td>Practical activites within the subject</td><td colspan="3">Not applicable</td></tr>
</table>

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