COURSE UNIT DESCRIPTION

Course unit title	Course unit code
Software Engineering Models and Methods	**PMKM7124**

Lecturer(s)	Department where the course unit is delivered
Coordinator: prof. dr. Romas Baronas **Other lecturers:** doc. Audronė Lupeikienė, doc. Karolis Petrauskas, lekt. Viktoras Golubevas	Department of Software Engineering Institute of Compute Science Vilnius University

Cycle	Level of course unit	Type of the course unit
Second	-	Compulsory

Mode of delivery	Semester or period when the course unit is delivered	Language of instruction
Face-to-face	2nd semester	Lithuanian, English

Prerequisites and corequisites	
Prerequisites: -	**Corequisites (if any): -**

Number of credits allocated	Student's workload	Contact hours	Self-study hours
10	260	80	180

Purpose of the course unit: programme competences to be developed		
To increase knowledge of models and methods of requirements engineering, software design, construction, testing and quality assurance, to increase expertise in the area of the conceptual and formal modeling, to develop skills in choosing and applying appropriate software engineering models, methods and tools in practice, to improve abstract thinking.		
Learning outcomes of the course unit: students will be able to	**Teaching and learning methods**	**Assessment methods**
Evaluate software engineering models, methods and tools and choose the right system for developing a specific application.	Lectures, problem-oriented teaching, case studies, information retrieval, literary reading, individual work, tutorials, laboratory work	Laboratory works and results presentation, written exam (open, semi-open and close-ended questions and tasks).
Formalize a subject area, to formally define the specific requirements and use them in the software development.		
Clearly present the chosen topic, summarize it, argue and defend his/her own opinion.	Information retrieval, literary readings, tutorials, report preparation and presentation at the seminar, group discussion, demonstration	Presentation material, the oral presentation, answers to oral questions

Course content: breakdown of the topics	Contact hours							Self-study work: time and assignments	
	Lectures	Tutorials	Seminars	Practice	Laboratory work	Practical training	**Contact hours**	**Self-study hours**	**Assignments**
1. The conception, purpose and classification of the software engineering models, methods and tools.	2						**2**	**2**	Self-study of literature.
2. Formal methods of software engineering, their classification,	2						**2**	**3**	Self-study of literature.

characteristics, application.									
3. Object constraint language (OCL) as an extension of UML. Kinds of constraints, structural and user-defined types, OCL expressions, OCL messages.	4				2		**6**	**15**	1st laboratory work: to create UML model for a chosen subject area (task), to define OCL requirements, and to generate program code using an appropriate tool. Self-study of literature.
4. An application of specific OCL constraints. Model-driven application development.	4				2		**6**	**15**	
5. Requirements modelling languages. Requirements modelling. Modelling quality aspects of the system. Requirements model analysis and requirements assessment.	6				4		**10**	**25**	2nd laboratory work: to create goals model for the chosen domain, to identify possible operationalizations for meeting non-functional requirements. Laboratory work: to create the requirements and their operationalizations model, to make model-based analysis to assess requirements consistency and feasibility. Self-study of literature.
6. Formal modeling with Z language. An implementation of models.	4				1		**5**	**10**	3rd laboratory work: to choose a formal software engineering method (Z system, algebraic specifications, JML or program analysis-synthesis) as well as a tool supporting the method, and to apply them to a self-chosen software development task (problem). Self-study of literature.
7. Description of the system behavior by algebraic specifications, their completeness and consistency. Prototyping the algebraic specifications.	6				1		**7**	**10**	
8. JML - Java modeling language. Static checking and runtime program checking.	4				1		**5**	**10**	
9. Formal program description by symbolic logic. Formal program analysis and synthesis by mechanical theorem proving.	4				1		**5**	**10**	
10. Modeling and verification of concurrent systems by applying a formal specification language TLA+ and tools.	4				4		**8**	**20**	4th laboratory work: to create and verify a model of a chosen concurrent system by TLA+ tool. Self-study of literature.
11. Specific models, methods and tools for software requirements, design, construction, testing and quality assurance.			24				**24**	**35**	Preparation of a presentation on a modern software engineering model, method and/or a corresponding tool (tools), which was not studied in the study program. Self-study of literature.
12. Preparing for the exam and taking the final exam (written)								**25**	Self-study of literature.
Total	**40**		**24**		**16**		**80**	**180**	

Assessment strategy	**Weight, %**	**Deadline**	**Assessment criteria**
Oral presentation	20	During the semester	The following aspects of the presentation are assessed: - The presentation structure, size and style: the structure is clear and logical, contains all necessary components (introduction, explanation, conclusions), the presentation is of a reasonable duration; the material was delivered for a preview - 1 point. - Completeness, recommendations and conclusions: The material presented in detail and in comparison, to other methods/tools, recommendations and conclusions are grounded - 1 points.
Four laboratory works	40	During the semester	The assessment of four laboratory works. For each fully completed and timely defended laboratory work, 1 point is awarded. If the work is done partially, in poor quality or late, the points are reduced.

			Lateness no more than 2 weeks leads to reducing the assessment in 25%, lateness no more than 4 weeks – 50%, later – 75%.
Exam (written)	40	Exam session	The exam consists of about 20 open, semi-open and close-ended questions and tasks each of them is assessed between 0.1 and 0.3 points. The questions are formulated from topics set out in lectures. The exam is allowed only after reading the report of the seminar, minimum two laboratory works and the collection of at least 3 points before the exam. The assessment of the exam: - 4 points: excellent knowledge and skills, the assessment level, collected at least 3.5 points; - 3 points: good knowledge and skills, the synthesis level, collected at least 2.5 points; - 2 points: average knowledge and skills, the analysis level, collected at least 1.5 points; - 1 point: knowledge and skills are less than average, the application level, collected at least 0.5 points.

Author	**Year**	**Title**	**Number or volume**	**Publisher or URL**
Required reading				
R. Baronas	2007	Software Engineering Methods and Tools (in Lithuanian)		Vilniaus universitetas, http://www.mif.vu.lt/~baronas/psmi/book
J. Warmer, A. Kleppe	2003	The Object Constraint Language: Getting Your Models Ready for MDA	2nd ed.	Addison-Wesley Professional
H. Habrias, M. Frappier	2000	Software Specification Methods: An Overview Using a Case Study (Formal Approach to Computing and Information Technology)		Springer-Verlag
S.L. Pfleeger	2009	Software Engineering: Theory and Practice	4th ed.	Prentice Hall
Recommended reading				
A. Kleppe, J. Warmer, W. Bast	2003	MDA Explained: The Model Driven Architecture-Practice and Promise		Addison-Wesley Professional
L. Lamport	2002	Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers		Addison-Wesley, Boston, MA, USA, https://lamport.azurewebsites.net/tla/book-02-08-08.pdf
C.-L. Chang, R.C.-T. Lee	1997	Symbolic Logic and Mechanical Theorem Proving		Academic Press
OMG	2014	UML 2.4 OCL Specification		Object Management Group, http://www.omg.org/spec/OCL
J.M. Spivey	2001	The Z Notation: A Reference Manual	3rd ed.	Oriel College, Oxford, http://spivey.oriel.ox.ac.uk/corner/Z_Reference_Manual
J.L. Turner, T.L. McCluskey	1992	The Construction of Formal Specifications: An Introduction to the Model-Based and Algebraic Approaches		http://helios.hud.ac.uk/scomtlm/book.pdf
K. Pohl	2010	Requirements Engineering. Fundamentals, Principles, and Techniques		Springer-Verlag