SYLLABUS

Methodologies for Software Processes

University year 2025-2026

1. Information regarding the programme

1.1. Higher education institution	Babeş Bolyai University
1.2. Faculty	Faculty of Mathematics and Computer Science
1.3. Department	Department of Computer Science
1.4. Field of study	Computer Science
1.5. Study cycle	Master
1.6. Study programme/Qualification	Security
1.7. Form of education	Full time

2. Information regarding the discipline

2.1. Name of the discipline	Methodologies for Software Processes					Discipline code	**MME8021**
2.2. Course coordinator					Assoc. Prof. Eng. Florin Craciun		
2.3. Seminar coordinator					Assoc.Prof. Eng. Florin Craciun		
2.4. Year of study	1	2.5. Semester	2	2.6. Type of evaluation	E	2.7. Discipline regime	Compulsory

3. Total estimated time (hours/semester of didactic activities)

3.1. Hours per week	**4**	of which: 3.2 course	**2**	3.3 seminar/laboratory/project	**2**
3.4. Total hours in the curriculum	56	of which: 3.5 course	28	3.6 seminar/laboratory/project	**28**

Time allotment for individual study (ID) and self-study activities (SA)	**hours**
Learning using manual, course support, bibliography, course notes (SA)	30
Additional documentation (in libraries, on electronic platforms, field documentation)	20
Preparation for seminars/labs, homework, papers, portfolios and essays	35
Tutorship	4
Evaluations	5
Other activities:	

3.7. Total individual study hours	94
3.8. Total hours per semester	150
3.9. Number of ECTS credits	6

4. Prerequisites (if necessary)

4.1. curriculum	none
4.2. competencies	Basic software development skills

5. Conditions (if necessary)

5.1. for the course	projector
5.2. for the seminar /lab activities	projector

6.1. Specific competencies acquired [1]

[1] One can choose either competences or learning outcomes, or both. If only one option is chosen, the row related to the other option will be deleted, and the kept one will be numbered 6.

Professional/essential competencies	• assimilation of mathematical concepts and formal models to understand, verify and validate software systems; • analysis, design, and implementation of software systems; • proficient use of methodologies and tools specific to programming languages and software systems;
Transversal competencies	• team work capabilities; able to fulfill different roles; • professional communication skills; concise and precise description, both oral and written, of professional results , negociation abilities;

6.2. Learning outcomes

Knowledge	The student knows: • The graduate has the necessary knowledge to devise, model and design of complex software application • The graduate possesses the fundamental knowledge for modelling, being able to analyse real life problems and to translate them in concrete requirements and to design a corresponding software model
Skills	The student is able to • The graduate has the ability to follow the entire life cycle of software system development • The graduate has the skills to perform research in the domain of educational sciences especially for algorithmic thinking and for critical thinking
Responsibility and autonomy:	The student has the ability to work independently to obtain • The graduate has the ability of interdisciplinary vision between computer science subdomains in order to combine them in a software system • The graduate can use specific language and terminology for software engineering being able to communicate and interact with members of a team

7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	• be able to apply basic methods for software formalization
7.2 Specific objective of the discipline	• Be able to write formal specifications • understanding of program verification • be able to use software verification tools

8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction in program verification	Exposure,description,	

	explanation, debate and dialogue, discussion of case studies	
2. Introduction in Viper concepts		
3.Foundational Reasoning Principles		
4.SMT solvers		
5. Building a first verifier		
6.Loops and procedures		
7.Advanced data types		
8. Heap and objects		
9.Abstractions in specifications		
10. Permission models		
11. Separation Logic		
12. Concurrency		
13. Concurrent Separation Logic		
14. Front-end verifiers		
Bibliography Viper tutorials and research papers https://www.pm.inf.ethz.ch/publications0.html		
8.2 Seminar / laboratory	Teaching methods	Remarks
1. Viper foundations	Use practical tools to implement group projects. Discuss research papers.	Seminar is organized as a total of 14 hours – 2 hours every second week Project is organized as a total of 14 hours – 2 hours every
2. Forward and Backward reasoning		
3. Z3 foundations		
4. Invariants and Recursion		
5. Advanced Data types		
6. Concurrency		
7. Concurrency		
8. Evaluation		
Bibliography Viper tutorials and research papers https://www.pm.inf.ethz.ch/publications0.html		

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

- The course respects the IEEE and ACM Curriculla Recommendations for Software Engineering studies;
- The content of the course is considered by the software companies as important for average software development skills

10. Evaluation

Activity type	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Percentage of final grade
10.4 Course	- know the basic principle of the domain; - apply the course concepts problem solving	Written exam	50.00%
10.5 Seminar/laboratory	- be able to implement course concepts – be able to use verification tools - be able to do a critical evaluation of research papers to be able to write a critical essay	-Practical assigments	50.00%
10.6 Minimum standard of performance			
• At least grade 5 (from a scale of 1 to 10) at both written exam and laboratory work.			

11. Labels ODD (Sustainable Development Goals)[2]

Not applicable.

[2] Keep only the labels that, according to the *Procedure for applying ODD labels in the academic process*, suit the discipline and delete the others, including the general one for *Sustainable Development* – if not applicable. If no label describes the discipline, delete them all and write „*Not applicable.*”.

Date:
...

Signature of course coordinator

Assoc.Prof.Eng. Florin Craciun

Signature of seminar coordinator

Assoc.Prof. Eng. Florin Craciun

Date of approval:
...

Signature of the head of department

Assoc.prof.phd. Adrian STERCA