SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babeş-Bolyai University**
1.2 Faculty	**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	**Inginerie Software / Software Engineering**

2. Information regarding the discipline

2.1 Name of the discipline	**Formal Methods in Programming**						
2.2 Course coordinator	**Lect. PhD. Vladiela Petraşcu**						
2.3 Seminar coordinator	**Lect. PhD. Vladiela Petraşcu**						
2.4. Year of study	**1**	2.5 Semester	**1**	2.6. Type of evaluation	**E**	2.7 Type of discipline	**Compulsory**

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

3.1 Hours per week	**3**	Of which: 3.2 course	**2**	3.3 seminar/laboratory	**1**
3.4 Total hours in the curriculum	**42**	Of which: 3.5 course	**28**	3.6 seminar/laboratory	**14**
Time allotment:					Hours
Learning using manual, course support, bibliography, course notes					**42**
Additional documentation (in libraries, on electronic platforms, field documentation)					**28**
Preparation for seminars/labs, homework, papers, portfolios and essays					**28**
Tutorship					**6**
Evaluations					**4**
Other activities:					
3.7 Total individual study hours	**108**				
3.8 Total hours per semester	**150**				
3.9 Number of ECTS credits	**6**				

4. Prerequisites (if necessary)

4.1. curriculum	Basic Computational Logic knowledge
4.2. competencies	-

5. Conditions (if necessary)

5.1. for the course	Videoprojector
5.2. for the seminar /lab activities	Computers, videoprojector, Internet acces

6. Specific competencies acquired

Professional competencies	**C 4.1** Ability to define fundamental computer science concepts and principles, as well as theories and mathematical models **C 4.2** Ability to interpret formal mathematical and computer science models **C 4.3** Ability to identify appropriate models and methods for solving real problems **C 4.5** Ability to incorporate formal models in specific applications from various fields
Transversal competencies	**CT1** Ability to apply rules of organized and efficient work, of a responsible attitude towards the teaching-scientific domain, in order to creatively harness one's own potential, by respecting the rules and principles of professional ethics **CT3** The use of effective methods and techniques of learning, information, research and capacity of knowledge exploitation, to adapt to a dynamic society and communication in Romanian and in an international language

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

7.1 General objective of the discipline	• Making students assimilate the fundamental aspects related to the use of formal techniques in system specification, development and verification, as well as making them acquire the ability to reason formally, to analyze algorithms correctness and to measure programs' quality.
7.2 Specific objective of the discipline	At the end of the course, students will: • be able to define Formal Methods (FMs), provide taxonomies and argue on the role and necessity of FMs in software development; • have knowledge of a full-fledged model-oriented formal method such as B and be able to (incrementally) specify a system in B, prove the consistency of the resulting specification and refine it; • understand the basic concepts related to model checking and be able to apply a model checking algorithm in order to verify whether a given system satisfies a particular property.

8. Content

8.1 Course	**Teaching methods**	**Remarks**
1. Introduction to Formal Methods. Overview of Formal Methods Techniques and Tools	Description, explanation, dialogue, examples	
2. Algorithm Correctness	Description, explanation, dialogue, examples, proofs	

3. Reliability-ensuring Paradigms	Description, explanation, dialogue, examples	
4. The B Method: Introduction to the Abstract Machine Notation (AMN) and Generalised Substitution Language (GSL)	Description, explanation, dialogue, examples, discussion of case studies, proofs	
5. B Mathematical Notation: Relations, Functions and Sequences	Description, explanation, dialogue, examples	
6. Structuring Mechanisms for B Specifications - INCLUDES	Description, explanation, dialogue, examples, discussion of case studies, proofs	
7. Structuring Mechanisms for B Specifications – SEES and USES	Description, explanation, dialogue, examples, discussion of case studies, proofs	
8. Refining B Specifications – Data Refinement	Description, explanation, dialogue, examples, discussion of case studies	
9. Refining B Specifications – Refinement of Nondeterminism and Proof Obligations for Refinement	Description, explanation, dialogue, examples, discussion of case studies, proofs	
10. From UML/OCL Models to B Specifications. Formal Verification of Object-Oriented Models	Description, explanation, dialogue, examples, discussion of case studies	
11. Introduction to Model Checking. System Modeling: Transition Systems	Description, explanation, dialogue, examples	
12. Property Specification: Temporal Logic	Description, explanation, dialogue, examples	
13. Model Checking Algorithms	Description, explanation, dialogue, examples	
14. Model Checking Tools	Description, explanation, dialogue, discussion of case studies	

Bibliography

[1] Abrial, J.-R., *The B Book - Assigning Programs to Meanings*, Cambridge University Press, 1996.
[2] Almeida, J.B., et al., *Rigorous Software Development: An Introduction to Program Verification*, Springer, 2011.
[3] Baier, C. and Katoen, J.-P., *Principles of Model Checking*, The MIT Press, 2008.
[4] Clarke, E.M. and Lerda, F., *Model Checking: Software and Beyond*, Journal of Universal Computer Science, 13(5), 639-649, 2007.
[5] Clarke, E.M., Wing, J.M., et al., *Formal Methods: State of the Art and Future Directions*, ACM Computing Surveys, 28(4):626-643, 1996.
[6] Frențiu, M. and Pop, H.F., *Fundamentals of Programming*, Cluj University Press, 2006. (chapter 2)
[7] Hall, A., Chapman, R., *Correctness by Construction: Developing a Commercial Secure System*, IEEE Software, January/February 2002, pp. 18–25.
[8] Haxthausen, A.E., *An Introduction to Formal Methods for the Development of Safety-critical Applications,* 2010.
[9] Holloway, C.M., *Why Engineers Should Consider Formal Methods*, Proceedings of the 16th Digital Avionics System Conference, 1997.
[10] McConnel, S., *Code Complete (2nd ed.)*, Microsoft Press, 2004.(Chapter 8 – Defensive Programming)

[11] Meyer, B., *Object-Oriented Software Construction (2nd ed.)*, Prentice-Hall, 1997. (Chapter 11 – Design by Contract: building reliable software)
[12] Meyer, B., *Applying „Design by Contract"*, IEEE Computer 25(10):40-51, 1992.
[13] Merz., S., *Model Checking: A Tutorial Overview*, Lecture Notes in Computer Science 2067, pp. 3 - 38, 2001.
[14] Mills, H., Dyer, M., Linger, R., *Cleanroom Software Engineering*, IEEE Software 4 (5): 19–25, 1987.
[15] Muler-Olm, M., Schmidt, D., and Steffen, B., *Model Checking: A Tutorial Introduction*, Lecture Notes in Computer Science 1694, pp. 330 - 354, 1999.
[16] Schneider, S., *The B-Method - An Introduction*, Palgrave Macmillan, Cornerstones of Computing series, 2001.

Links:

[1] Clearsy System Engineering, *AtelierB home page* http://www.atelierb.eu/en/
[2] Clearsy System Engineering, *B Method home page* http://www.methode-b.com/en/

8.2 Seminar	**Teaching methods**	**Remarks**
1. Organizing Moments. Myths and Commandments of Formal Methods. Industrial FM Success Stories	Description, explanation, conversation, debate, case studies	Two seminar hours every other week
2. Proving Algorithm's Correctness. Paper Presentations.	Description, explanation, conversation, debate, examples, proofs	
3. Introduction to the AtelierB tool. Simple Abstract Machine Specifications and Consistency Checks using AtelierB Paper Presentations.	Description, explanation, conversation, debate, examples, proofs	
4. Incremental B Specification Examples Paper Presentations	Description, explanation, conversation, debate, examples, proofs	
5. B Refinement Examples Paper Presentations	Description, explanation, conversation, debate, examples, proofs	
6. Model Checking Examples Paper Presentations	Description, explanation, conversation, debate, examples, proofs	
7. Project Presentations	Description, explanation, conversation	

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

- This course follows the IEEE and ACM Curriculla Recommendations for Software Engineering studies;
- Courses with similar content are taught in the major universities in Romania offering similar study programs;
- Course content is considered important by the software companies, for improving the reliablity of the resulting software products.

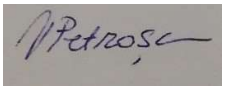
10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	- knowledge of the fundamental FM concepts and techniques taught during lectures - ability to specify a basic system in B and manually prove its consistency - ability to apply a model checking algorithm on a simple example	Written exam	50%
10.5 Seminar/lab activities	- ability to select a relevant FM paper - ability to summarize its contents in a written paper report - ability to present the paper in a talk and sustain a debate around its enclosed ideas	Scientific FM paper presentation	25%
	- ability to formally specify a given system in AtelierB - ability to use the tool for type checking the specification, generate the associated proof obligations and discharge them	B Project: B specification and consistency checking using AtelierB of an informally specified system	25%
10.6 Minimum performance standards			
Grade at least 5 (from a scale of 1 to 10) at written exam, paper and project.			

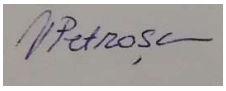
Date

28 aprilie 2023

Signature of course coordinator



Signature of seminar coordinator



Date of approval

.....

Signature of the head of department

Prof. PhD. Laura Dioșan

.....