SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babes-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	**Bachelor**
1.6 Study programme / Qualification	**Computer Science**

2. Information regarding the discipline

2.1 Name of the discipline	**Software Systems Verification and Validation**						
2.2 Course coordinator	**PhD Lecturer Vescan Andreea**						
2.3 Seminar coordinator	**PhD Lecturer Vescan Andreea**						
2.4. Year of study	**3**	2.5 Semester	6	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	4	Of which: 3.2 course	2	3.3 seminar/laboratory	2
3.4 Total hours in the curriculum	48	Of which: 3.5 course	24	3.4 seminar/laboratory	24
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					24
Additional documentation (in libraries, on electronic platforms, field documentation)					24
Preparation for seminars/labs, homework, papers, portfolios and essays					24
Tutorship					6
Evaluations					24
Other activities:					0

3.7 Total individual study hours	102
3.8 Total hours per semester	150
3.9 Number of ECTS credits	6

4. Prerequisites (if necessary)

4.1. curriculum	•
4.2. competencies	•

5. Conditions (if necessary)

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

6. Specific competencies acquired

Professional competencies	• Knowledge, understanding and use of basic concepts of theoretical Computer Science • Ability to work independently in order to solve problems in defined professional contexts.
Transversal competencies	• Improved programming abilities: debugging and correcting compilers errors

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

7.1 General objective of the discipline	• To understand what a correct algorithm is. • To gain knowledge of designing correct algorithms and proving their correctness hand- in-hand. • To learn the methods of program verification and validation. • To become used with building correct programs from specifications. • To acquire a modern programming style.	
7.2 Specific objective of the discipline	• Students will know how and which are the steps of an inspection, either of the source code or specification of each stage of the development of the software system. • Students will know to create from the specification and design phase test cases that will help them develop a better and robust software system.	•

8. Content

8.1 Course	Teaching methods	Remarks
1. Verification and validation (the concepts verification and validation);	Presentation, Didactic demonstration, Problematizations	
2. Program testing (1): the concept of Program testing; unit testing: testing criteria, blackbox and whitebox testing;	Presentation, Didactic demonstration, Problematizations	
3. Program testing(2): types of testing(integration T., system T., regression T., acceptance T.), testing automatizing;	Presentation, Didactic demonstration, Problematizations	
4. Program inspection	Presentation, Didactic demonstration, Problematizations	
5. Symbolic execution	Presentation, Didactic demonstration, Problematizations	
6. Model checking	Presentation, Didactic demonstration, Problematizations	
7. The theory of program correctness. The evolution of	Presentation, Didactic	

the concept of program correctness. The Contribution of Floyd, Hoare, Dijkstra, Gries, Droomey, Morgan	demonstration, Problematizations	
8. Program Specification. Floyd's method for prooving correctness. Dijkstra: the weakest precondition. Stepwise refinement from specifications Hoare's axiomatisation method	Presentation, Didactic demonstration, Problematizations	
9. Comparing the verification methods (correctness-inspection-testing-symbolic execution) Verification and validation: How? Who? When?	Presentation, Didactic demonstration, Problematizations	
10. Cleanroom. Program Quality.	Presentation, Didactic demonstration, Problematizations	
11. Quality, SPI, SQA,CMM.	Presentation, Didactic demonstration, Problematizations	
12. The consequences of the theory of program correctness on programming. Programming style.	Presentation, Didactic demonstration, Problematizations	

Bibliography

1. BALANESCU T., Corectitudinea programelor, Editura tehnica, Bucuresti 1995.
2. DIJKSTRA, E., A constructive approach to the problem of program correctness, BIT, 8(1968), pg.174-186.
3. DIJKSTRA, E., Guarded commands, nondeterminacy and formal derivation of programs, CACM, 18(1975), 8, pg.453-457.
4. DROMEY G., Program Derivation. The Development of Programs From Specifications, Addison Wesley Publishing Company, 1989.
5. FRENTIU, M., Verificarea corectitudinii programelor, Ed.Univ."Petru-Maior", 2001.
6. GRIES, D., The Science of Programming, Springer-Verlag, Berlin, 1981.
7. HOARE, C.A.R., An axiomatic basis for computer programming, CACM, 12(1969), pg.576-580, 583.
8. Morgan, C., Programing from Specifications, Prentice Hall, NewYork, 1990.

B. Internet

8.2 Seminar / laboratory	Teaching methods	Remarks
1. S1: Specifications and Inspection L1: Static analysis using ESCJava2, JML	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
S2: Test cases using Black-box Testing (BBT) and White-box (WBT)testing techniques L2:Black-box Testing	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
S3: Control paper 1 + BBT and WBT L3:White-Box Testing	Presentation, Conversation,	

	Problematizations, Discovery, Other methods – individual study, exercises	
S4: Model checking+ Correctneses L4:Model cheking	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
S5: Control paper 2 +Correctness L5:Testing GUI, Web app.	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
S6:Inspection L6:Inspection	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
Bibliography		

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

- Students will know how to apply testing methods for a software products, testing methods that are used in industry.
- Students will learn various verification and validation methods of a software system.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	At the end a written examination will give a mark E.	Written examination	50
10.5 Seminar/lab activities	The activity at seminaries, consisting from participation in solving the exercises and discussions, will be appreciate by a mark S.	Control paper 1+ Control paper 2+ Seminar activity	25
	A second mark L will be given for the laboratories work.		25
10.6 Minimum performance standards			
➢ Students will learn and apply testing methods for a software product.			

- Students will apply various methods for verification (testing, inspection, model checking) for establishing the
- correctness of an algorithm.
- At least grade 5 (from a scale of 1 to 10) at written exam and laboratory work and seminar activity.

Date	Signature of course coordinator	Signature of seminar coordinator
04.30.2013	Lect. PhD. Andreea Vescan,	Lect. PhD. Andreea Vescan

Date of approval	Signature of the head of department
.....	Prof. PhD. Bazil Parv