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	3.6 seminar/laboratory	24

Time allotment:	hours
Learning using manual, course support, bibliography, course notes	28
Additional documentation (in libraries, on electronic platforms, field documentation)	28
Preparation for seminars/labs, homework, papers, portfolios and essays	28
Tutorship	6
Evaluations	12
Other activities:	-

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	• Identification of proper methodologies for software systems development; • Identification and explication of proper software systems specification methods; • Using methodologies and tools for development of informatics applications; • Using proper criteria and methods for evaluation of software applications; Realization of dedicated information projects.
Transversal competencies	• Application of efficient and rigorous working rules, manifest responsible attitudes toward the scientific and didactic fields, respecting the professional and ethical principles. • Use of efficient methods and techniques for learning, information, research and development of abilities for knowledge exploitation, for adapting to the needs of a dynamic society and for communication in Romanian as well as in a widely used foreign language

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. • Students will know how to use tools for the management of testing process. • Students will know how to design test cases using various criteria (white-box, black-box).

8. Content

8.1 Course	Teaching methods	Remarks
1. Verification and validation (the concepts verification and validation); Quality Assurance and Quality Control	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
2. Program testing (1): the concept of Program testing; unit testing: testing criteria, blackbox and whitebox testing;	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
3. SPI, SQA,CMM. Cleanroom. Program Quality.	• Interactive exposure • Explanation • Conversation • Didactical demonstration	

4. Program testing(2): types of testing(integration T., system T., regression T., acceptance T.), testing automatizing;	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
5. Testing GUI	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
6. Testing Web applications. Selenium Web Driver	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
7. Program inspection	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
8. Symbolic execution	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
9. Model checking	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
10. The theory of program correctness. The evolution of the concept of program correctness. The Contribution of Floyd, Hoare, Dijkstra, Gries, Droomey, Morgan	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
11. Program Specification. Floyd’s method for prooving correctness. Dijkstra: the weakest precondition. Stepwise refinement from specifications Hoare’s axiomatisation method	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
12. Comparing the verification methods (correctness-inspection-testing-symbolic execution) Verification and validation: How? Who? When?	• Interactive exposure • Explanation • Conversation • Didactical demonstration	

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
Seminar 1: • Test cases using Black-box Testing (BBT) Laboratory 1: • Test cases using Black-box Testing (BBT) • Test management tool (e.g. Testlink) • Issue traker tool (e.g. Bugzilla)	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
Seminar 2: • Test cases using White-box Testing (WBT) Laboratory 2: • Test cases using White -box Testing (WBT) • Test management tool (e.g. Testlink) • Issue traker tool (e.g. Bugzilla)	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
Seminar 3: • Levels of testing Laboratory 3: • Levels of testing • Test management tool (e.g. Testlink) • Issue traker tool (e.g. Bugzilla) • Continuous Integration tool (Jenkins)	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
Seminar 4: • Control paper: WBT+BBT - test cases • Inspection Laboratory 4: • Inspection • Inspection tool • Issue traker tool (e.g. Bugzilla)	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
Seminar 5: • GUI/Web testing Laboratory 5: • GUI/Web testing • Web testing tool (e.g. Selenium Web Driver) • Issue traker tool (e.g. Bugzilla)	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
Seminar 6: • Correctness Laboratory 6: Static analysis using ESCJava2, JML	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 10.5 Seminar/lab activities	At the end a written examination will give a mark E.	Written examination	50
	• 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

04.30.2014 Lect. PhD. Andreea Vescan, Lect. PhD. Andreea Vescan

Date of approval Signature of the head of department

.... Prof. PhD. Bazil Parv