COURSE SHEET

Software Engineering

Academic year 2022-2023

1. About the program

1.1	University	Universitatea din Piteşti
1.2	Faculty	Sciences, Physical Education and Computer Science
1.3	Department	Mathematics-Computer Science
1.4	Field of study	Informatics
1.5	Cycle of studies	Master
1.6	Study Program / Qualification	Advanced techniques for information processing/ Advanced techniques for information processing

2. Discipline data

2.1	Name of the discipline					*Software Engineering*					
2.2	The holder of the course activities					Tudor Bălănescu					
2.3	Holder of laboratory activities					Tudor Bălănescu					
2.4	Year of study	1	2.5	Semester	1	2.6	Type of assessment	E	2.7	Discipline regimen	O

3. Estimated total time

3.1	Number of hours per week	4	3.2	of which course	2	3.3	laboratory	2
3.4	Total hours of the curriculum	56	3.5	of which course	28	3.6	laboratory	28
Distribution of the time fund								hours
Study by textbook, course support, bibliography and notes								56
Additional documentation in the library, on specialized electronic platforms and in the field								38
Preparation of seminars/ laboratories, themes, papers, portfolios, essays								40
Tutoring								6
Examination								4
Other activities.....								-

3.7	Total hours of self-study	144
3.8	**Total hours per semester**	**200**
3.9	**Number of credits**	**8**

4. Preconditions (where applicable)

4.1	Curriculum	-
4.2	Skills	-

5. Conditions (where applicable)

5.1	Conduct of the course	Room with video projector
5.2	Conducting the seminar/laboratory	Room with video projector and computer equipment

6. Acquired specific skills

Professional skills	Ability to specify software systems and user requirements. Skills to develop formal odels of software systems using languages and formalisms like UML, Z, Petri nets, state charts, FSMs, timed automata. Knowledge of theoretical procedures for automated verification of models and validation of implementations . Realization of projects..
Transversal competences	Applying the rules of organized and efficient work, of responsible attitudes towards the scientific-professional field, for the creative capitalization of one's own potential, respecting the principles and norms of professional ethics; Efficiently carrying out the activities organized in an interdisciplinary team by assuming execution and leadership functions, with the development of empathic capacities of inter-personal communication, networking and collaboration with various groups; Elaboration of own professional development project; the use of effective methods and techniques for learning, information, research and capacity development, for valuing knowledge, for adapting to the requirements of a dynamic society and for communicating in Romanian and English.

7. The objectives of the discipline

7.1 The general objective of the discipline	▶ The discipline has as general objective the acquisition by students of the basic knowledge for formal specification design, implementation, verification and validation of software systems.
7.2 Specific objectives	Cognitive objectives: ▶ . Learning and mastering the basic concepts in the discipline of "software engineering".. *Procedural objectives:* ▶ Development of a software development plan, going through all the phases from requirements development, system modeling, implementation and testing, use of software packages and products that support the above activities, especially UML language modeling, application steps above

	Attitudinal objectives: ▶ Rigor in the specification, design, implementation, verification and validation of software systems..

8. Contents

8.1. Course		Nr. hours	Teaching methods	Observations Resources used
1	Introduction • Professional software development • Software engineering ethics Case studies	2	lecture problematization algorithms debate individual themes group work Explanation Description and exemplification Demonstration Heuristic Conversation Exercise	computer projector
2	Software Processes • Software process models • Process activities • Coping with change The Rational Unified Process	2		
3	Agile software development • Agile methods • Plan-driven and agile development • Extreme programming • Agile project management Scaling agile methods	2		
4	Requirements engineering • Functional and non-functional requirements • The software requirements document • Requirements specification • Requirements engineering processes • Requirements elicitation and analysis • Requirements validation • Requirements management	2		
5	System modeling • Context models • Interaction models • Structural models • Behavioral models • Model-driven engineering	6		
6	Architectural design • Architectural design decisions • Architectural views • Architectural patterns • Application architectures	4		
7	Design and Implementation • Object-oriented design using the UML • Design patterns • Implementation issues • Open source development	4		
8	Software testing – 4 ore • Development testing • Test-driven development • Release testing • User testing • Unit, Integration, System testing	4		
9	Other topics – 2 ore • Software Evolution • Software Reuse • Distributed Software Engineering • Service-oriented Architecture	2		

Bibliography

1. Ian Sommerville. Software Engineering – 9th edition (2010) - Addison-Wesley
2. G. Booch, J. Rumbaugh, I. Jacobson. The Unified Language User Guide, Addison-Wesley, 1999.
3. Aditya P. Mathur, Foundation of Software Engineering, Dorling Kindersley, 2008.
4. Paul Ammann, Jeff Offutt. Cambridge University Press, 2008.
5. M. Fowler, K. Scott. UML Distilled: Applying the Standard Object Modeling Language, Addison-Wesley, 1997.
6. F. Ipate. Modelare orientata pe obiecte cu UML, Editura Universitatii Pitesti, 2001.
7. M. Roper. Software Testing, McGraw-Hill, 1994.
8. M. Holcombe, F. Ipate. Correct Systems: Building a Business Process Solution, Springer Verlag, 1998.

9. Tudor Bălănescu, Horia Georgescu, Marian Gheorghe, Peter O'Donogue: "HOOD and Regular Expressions", Analele Universităţii Bucureşti, Seria Matematică-Informatică, Special Issue, Proceedings of the Annual Meeting of the Faculty of Mathematics, 28-39 Nov. 1996, p. 45-60, 1997.

8.2. Applications – Seminar / Laboratory		Nr. hours	Teaching methods	Observations Resources used
1	**Software project that faithfully reflects the notions and stages of software development taught in the course • Examples for the notions taught in the course • Presentation of UML modeling tools (eg Magic Draw, ArgoUML)**	4	Explanation Description and exemplification Case study Exercise Problematization Individual themes Group work Debate	Computers. Software tools (IDEs)
2	**Specification of the requirements for use and development of a project, described in UML**	4		
3	**Use charts, state charts, sequence charts**	4		
4	**Example of a formal specification of a real time system**	4		
5	**Formal specification of properties and model checking**	4		
6	**Functional testing**	4		
7	**Structural testing**	4		

Bibliography

1. Ian Sommerville. Software Engineering – 9th edition (2010) - Addison-Wesley
2. F. Ipate. Modelare orientata pe obiecte cu UML, Editura Universitatii Pitesti, 2001.
3. Aditya P. Mathur, Foundation of Software Engineering, Dorling Kindersley, 2008.
4. Paul Ammann, Jeff Offutt. Cambridge University Press, 2008.

9. Corroborating the contents of the discipline with the expectations of the representatives of the epistemic community, professional associations and employers in the field related to the program

The competences acquired within the discipline allow the graduates to efficiently use formal methodologies for design, implementation, verification and validation of software systems.

10. Evaluation

Activity Type	10.1 Assessment criteria	10.2 Assessment methods	10.3 Percent of final grade
10.4 Course	Final evaluation	Practical test (algorithms and problems)	50%
10.5 Seminar/ Laboratory	Activity (solving proposed problems) Homework	Verification of solutions, practical test Homework check	30% 20%
10.6 Minimum performance standard	* Marks of at least 5 for the laboratory activity, for the homework and for the final evaluation (50% solving the requirements); final grade at least 5. * Set of minimal knowledge for passing the final exam: - Knowledge of the main computational models studied; Knowledge of ways of adequate application and efficient implementation of these models in solving the proposed problems.		

Date of completion 23.09.2022

Course holder
Tudor Balanescu

Laboratory holder
Tudor Balanescu

Date of approval in the Department

Director Department (provider)
Conf.univ.dr. Doru CONSTANTIN

Director Department (*beneficiary)*)
Conf.univ.dr. Doru CONSTANTIN