SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	30.

2. Data about the subject

2.1	Subject name					Design with Microprocessors					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Conf. dr. eng. Tiberiu Marita – tiberiu.marita@cs.utcluj.ro					
2.4	Teachers in charge of applications					Conf. dr. eng. Tiberiu Marita – tiberiu.marita@cs.utcluj.ro Sl. dr. eng. Mihai Negru – mihai.negru@cs.utcluj.ro As. drd. eng. Razvan Itu - Razvan.Itu@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Design with Microprocessors**	**2**	**-**	**1**	**1**	**28**	**-**	**14**	**14**	**74**	**130**	**5**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	28
Supplementary study in the library, online and in the field	14
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	28
Tutoring	0
Exams and tests	4
Other activities	0

3.7	Total hours of individual study	74
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. Pre-requisites (where appropriate)

4.1	Curriculum	Computer Architecture, Computer Programming
4.2	Competence	Hardware design, Assembly language programming, C language programming

5. Requirements (where appropriate)

5.1	For the course	Black-board/ White-board, projector, computer
5.2	For the applications	Computer, Atmel Studio, Arduino IDE, Arduino & RPi development boards, Pmods and several other components, modules, sensors etc.

6. Specific competences

Professional competences	**C2** – Designing hardware, software and communication components (2 credits) **C2.1 -** Describing the structure and operation of hardware, software and communication components **C2.2 -** Explaining the role, interaction and operation of hardware, software and communication components **C2.3 -** Construction of hardware and software components of computing systems using design methods, languages, algorithms, data structures, protocols and technologies **C2.4 -** Metric based evaluation of functional and non-functional characteristics of computing systems **C2.5 -** Implementation of hardware, software and communication components **C5** - Designing, managing the lifetime cycle, integrating and ensuring the integrity of hardware, software and communication systems (3 credits) **C5.1 -** Specifying the relevant criteria regarding the lifetime cycle, quality, security and computing system's interaction with the environment and human operator **C5.2 -** Using interdisciplinary knowledge for adapting an information system to application domain requirements **C5.3 -** Using fundamental principles and methods for security, reliability and usability assurance of computing systems **C5.4 -** Adequate utilization of quality, safety and security standards in information processing **C5.5 -** Realization of a project including problem identification and analysis, design and development, while proving the understanding of the basic quality needs and requirements
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Knowledge, understanding and use of concepts like microprocessor/ microcontroller, bus, memory system, data transfer methods, interface circuits and peripheral devices interfacing, analysis and design of microprocessor systems.
7.2	Specific objectives	To achieve the main objective, specific objectives are pursued: • Knowledge of microprocessors and microcontrollers features and capabilities: hardware capabilities, instruction set architecture, assembly language, and programming solutions. • Knowledge of hardware components used with the microprocessors: electrical and logical characteristics, connection modes. • Development of skills to find solutions based on microprocessors or microcontrollers for real problems with average complexity. • Acquaintance with microcontroller development boards and their software programming tools.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Lecture Overview. Introduction to MP based systems (AVR MCU family)	Oral, blackboard and multimedia, interactive teaching style, consultations, involvement of students in research / design.	N/A
2	AVR registers and instructions		
3	AVR I/O ports and interrupts		
4	Input/output and interrupts for Arduino systems		
5	AVR timers. Timing events with Arduino		
6	Serial data communication. Serial data transfer with Arduino		
7	Analog signals processing		
8	Microcontroller based applications: usage of sensors		
9	Microcontroller based applications: usage of actuators		
10	Introduction to the 8086 microprocessor family		
11	I/O transfer		
12	Memory interfacing		
13	Simple I/O interfaces. Parallel interfaces		
14	Multiprocessor systems		

Bibliography

1. B. B. Brey, "INTEL Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Prentium ProProcessor, Pentium II, III, 4", ed. 7, Prentice Hall, 2005

2. S. Nedevschi, “Microprocesoare”, Editura UTCN, 1994.
3. M.A. Mazidi,S. Naimi, S. Naimi, AVR Microcontroller and Embedded Systems: Using Assembly and C, Prentice Hall, 2010, ISBN 9780138003319.
4. M. Margolis, Arduino Cookbook, 2-nd Edition, O’Reilly, 2012.
Online: http://users.utcluj.ro/~tmarita/PMP/PMPcurs.htm

8.2. Applications (Laboratory, Projects)		Teaching methods	Notes
Laboratory			
1	Introduction to the Arduino boards.	Presentation on the blackboard, experiments on microcontroller development boards (Cerebot) use of specialized IDE design tools (AVR Studio), involvement of students in research / design.	N/A
2	Applications with simple I/O modules		
3	Working with the LCD shield and the interrupt system		
4	Usage of timers		
5	Communication interfaces		
6	Digital sensors. Analogue keypad		
7	Analogue signals processing.		
8	The "Processing" environment		
9	Usage of DC- and servo-motors		
Projects			
1	Project specification and study		
2	Project logic design		
3	Project implementation		
4	Project optimization, testing and validation		
5	Project assessment.		

Bibliography
1. Atmel ATmega2560 - 8 bit AVR Microcontroller datasheet, http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf
2. Arduino Mega 2560, http://arduino.cc/en/Main/ArduinoBoardMega2560
3. Abdul Maalik Khan, AVR Project Book, http://www.digisoft.com.pk/products/avr-project-book
4. Mike McRoberts, Beginning Arduino, 2-nd Edition, Technology in Action.
5. M. Margolis, Arduino Cookbook, 2-nd Edition, O’Reilly, 2012.
Online: http://users.utcluj.ro/~tmarita/PMP/PMPcurs.htm

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The course is in the Computer and Information Technology field. Its contents combine fundamentals with specific aspects of the used hardware and software tools, accustoming students with the design principles for microprocessor based systems. The course content was discussed with other universities in the country and abroad, and in conjunction with products /development tools offered by companies in Romania, Europe and the USA (e.g. Digilent, Atmel, Arduino, RaspberyPi) and is rated by the Romanian government agencies (CNEAA and ARACIS).

10. Evaluation

Activity type	10.1 Assessment criteria	10.2 Assessment methods	10.3 Weight in the final grade
Course	Testing theoretical knowledge and problem solving skills	Written exam	50 %
Applications	Practical skills for problem solving and implementation of specific problems for applications design. Attendance and activity	Colloquium, lab. work and project evaluation	50 %
10.4 Minimum standard of performance			
Modeling and implementation of typical engineering problems using the theoretical models and applicative tools specific to the domain.			

Course responsible
Assoc.prof.dr.eng. Tiberiu Marita

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	31.

2. Data about the subject

2.1	Subject name					Logic Programming					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof. dr. eng. Rodica Potolea – Rodica.Potolea@cs.utcluj.ro					
2.4	Teachers in charge of applications					Sl. dr. eng. Camelia Lemnaru – Camelia.Lemnaru@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Logic Programming**	**2**	**1**	**2**	**-**	**28**	**14**	**28**	**-**	**60**	**130**	**5**

3.1	Number of hours per week	5	3.2	of which, course	2	3.3	applications	3
3.4	Total hours in the teaching plan	70	3.5	of which, course	28	3.6	applications	42

Individual study	Hours
Manual, lecture material and notes, bibliography	28
Supplementary study in the library, online and in the field	10
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	14
Tutoring	3
Exams and tests	5
Other activities	0

3.7	Total hours of individual study	60
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. Pre-requisites (where appropriate)

4.1	Curriculum	Fundamental Algorithms, Programming
4.2	Competence	Logic

5. Requirements (where appropriate)

5.1	For the course	Whiteboard, projector, computer
5.2	For the applications	Computers, specific software (SICStus Prolog). Mandatory attendance of seminars and laboratory works.

6. Specific competences

Professional competences	**C2** Designing hardware, software and communication components (5 credit points) **C2.1** Describing the structure and functioning of computational, communication and software components and systems **C2.2** Explaining the role, interaction and functioning of hardware, software and communication components **C2.3** Building the hardware and software components of some computing systems using algorithms, design methods, protocols, languages, data structures, and technologies **C2.4** Evaluating the functional and non-functional characteristics of the computing systems using specific metrics **C2.5** Implementing hardware, software and communication systems
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	The main goal of the topic is getting the ability of symbolic processing in general, and logic processing in particular; moreover, acquiring abilities for providing specifications in logic, executable form. Estimating the performance of the solutions designed and implemented in logic formalism.
7.2	Specific objectives	Declarative and procedural semantics Extra-logic operators Meta-programming Data Structures in logic programming. techniques associated with efficiency estimation Incomplete structures, difference lists Types of recursions with advantages and limitations Development of complex applications

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction, first order logic declarative and procedural semantics	Interactive Course. Teaching relying on examples, questions and discussions. Continuous evaluation of knowledge aquisition.	
2	First order logic declarative and procedural semantics (continued)		
3	Negation as failure; Backtracking and cut		
4	Prolog programming techniques		
5	Prolog programming techniques (continued)		
6	Prolog programming techniques (continued)		
7	Prolog programming techniques (continued)		
8	Metalogic predicates		
9	Extra-logic predicates		
10	Nondeterministic Programming		
11	Incomplete data structures; difference lists		
12	Search techniques		
13	Search techniques (continued)		
14	Search techniques (continued)		

Bibliography

5. L. Sterling, E. Shapiro, *The Art of Prolog*, MIT Press, 1994.
6. W.F. Clocksin, C.S. Mellish , *Programming în Prolog*, Springer-Verlag Telos, 1994.
7. R. Potolea, *Programare Logică,* vol 1,U.T.Pres, 2007.

8.2. Applications (Seminars, Laboratory)		Teaching methods	Notes
1	Prolog language	Semiras and hands on laboratory works with specific topics. Problem solving with tracing and performance evaluation.	Seminars – design solutions to problem, implementation on board. Laboratory -
2	Sets, sorting		
3	Lists		
4	Basic operations on lists		
5	Incomplete lists; difference lists		
6	Trees		
7	Searching in trees		

8	Incomplete trees		computer work. (individual)
9	Modeling control structures in Prolog		
10	Graphs		
11	Searching in graphs		
12	Basic graphs algorithms		
13	Metaprogramming		
14	Hands on evaluation	Hands on evaluation	mandatory

Bibliography

1. Rodica Potolea, Programare Logica, UT Pres, 2007
2. T.Muresan, R. Potolea, C. Lemnaru, Resources for the laboratory sessions http://users.utcluj.ro/~cameliav/lp.php
3. T. Mureşan, R. Potolea, E. Todoran, A.D. Suciu, *Programare Logică - Indrumător de Laborator*, Romsver, 1998.

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

Classical topic of the Computer Science and Information Technology domain, which develops the ability to express executable specifications in a logic language (standard Prolog, Sictus Prolog). The topic enables the assimilation of knowledge and builds necessary skills to other disciplines (AI family), and useful in fundamental / applied research. Ability to analyze specifications and solutions in a unified manner, following partial and total correctness and efficiency at the same time.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Problem solving using specific techniques		Partial Exam (PE) (written) + Final Exam (FE) (written and / or oral)		20% +50%
Applications		Problem solving		Practical test (Lab) (PC)		30%

10.4 Minimum standard of performance

Grade=0.2*PE+ 0.3*Lab+ 0.5*FE. The condition for participation in the final exam: Lab> = 5. Condition promotion Grade > = 5

Course responsible
Prof.dr.eng. Rodica Potolea

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. **Data about the program of study**

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	32.

2. **Data about the subject**

2.1	Subject name					Functional Programming					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Lect. dr. eng. Octavian Pop – Octavian.Pop@cs.utcluj.ro					
2.4	Teachers in charge of applications					Lect. dr. eng. Octavian Pop – Octavian.Pop@cs.utcluj.ro Lect. dr.Camelia Chira – Camelia.Chira@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. **Estimated total time**

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Functional Programming**	**2**	**-**	**2**	**-**	**28**	**-**	**28**	**-**	**74**	**130**	**5**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28
Individual study								Hours
Manual, lecture material and notes, bibliography								28
Supplementary study in the library, online and in the field								14
Preparation for seminars/laboratory works, homework, reports, portfolios, essays								28
Tutoring								4
Exams and tests								
Other activities								

3.7	Total hours of individual study	74
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. **Pre-requisites (where appropriate)**

4.1	Curriculum	Data Structures and Algorithms Course
4.2	Competence	This course assumes no prior knowledge of functional programming, but advises at least one year of programming experience in a regular programming language such as Java, C, C++.

5. **Requirements (where appropriate)**

5.1	For the course	Basic notions of programming
5.2	For the applications	Linux

6. Specific competences

Professional competences	**C2** Designing hardware, software and communication components (4 credits) **C2.1** Describing the structure and functioning of computational, communication and software components and systems **C2.2** Explaining the role, interaction and functioning of hardware, software and communication components **C2.3** Building the hardware and software components of some computing systems using algorithms, design methods, protocols, languages, data structures, and technologies **C2.4** Evaluating the functional and non-functional characteristics of the computing systems using specific metrics **C2.5** Implementation of hardware, software and communication components
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Increasing the ability to develop more correct and concise code
7.2	Specific objectives	Writing better code with the concepts introduced by functional programming: high order functions, lazy evaluation, lambda calculus, infinite structure.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction. Programming Paradigms	Slides, Various student engagement techniques New examples Quick individual work (1 minute) Homework after each class discussed at the beginning of the next class.	
2	Basic concepts of programming in Hugs, ML, Lisp: functions, constants, primitive data types, recursion, tuples, infix operators, evaluation.		
3	Basic concepts: local declarations, polymorphism.		
4	Lists: list construction, basic operations on lists.		
5	Lists: polymorphic equality.		
6	Lists: list operators (generators, filters, list expressions).		
7	Trees: alternative data, pattern matching, exceptions, binary trees (list-tree conversions).		
8	Trees: binary trees (binary search trees, AVL balanced trees, examples (operations on sets)).		
9	Trees: binary trees (examples (Huffman codes)), propositional reasoner (example).		
10	Higher-order functions: anonymous functions, partial application, functions as data, data as functions, combinator functions, functionals for lists (list operator style, style without lists).		
11	Infinite data: lazy evaluation, unbounded objects, circular structures.		
12	Transformation and reasoning: structural induction, equivalence of functions, structural induction on trees, induction on number of nodes, general principle of induction.		
13	Lambda calculus: Lambda notation, conversions, combinators.		
14	Para-functional programming: basic language, mapped expressions, eager expressions.		

Bibliography

1. Haskell - A Purely Functional Language, http://www.haskell.org/
2. I.A. Leţia, Programare funcţională, Ed. UTPres, UTCN, 1996.
3. H. Conrad Cunningham, Notes on Functional Programming with Haskell, 2007
4. Raul Rojas, A Tutorial Introduction to the Lambda Calculus, FU Berlin, WS-97/98

8.2. Applications (Laboratory)		Teaching methods	Notes
1	Lisp objects, form evaluation, primitive Lisp functions.	New examples Tracing algorithms Midterm assessment	
2	Internal representation, control of evaluation, function definition. Recursion and iteration.		
3	Scope of variables, iterative forms. LAMBDA-expressions, higher-order functions, mapping.		

4	Association lists, properties, arrays and structures. Macrodefinitions, functions as data, surgery.	Miniprojects	
5	Trees in Lisp. Graphs and backtracking.		
6	Pattern matching. Symbolic processing.		
7	Lisp microinterpreter. Review of programming in Lisp, in preparation for the lab test.		
8	Lab test (Programming in Lisp).		
9	ML Lists, Recursion,.		
10	ML type checking		
11	ML Trees		
12	Haskell – High order functions		
13	Haskell -Lazy evaluation, circular lists, infinite lists.		
14	Lab test (Programming in ML and Haskell).		

Bibliography

1. I.A. Leţia, E.Şt. Chifu, C. Cenan, Programare funcţională. Îndrumător de laborator, Ed. Casa cărţii de ştiinţă, 1999.
2. David S. Touretzky, Common Lisp: A Gentle Introduction to Symbolic Computation, The Benjamin/Cummings Publishing Company,Inc, 1989
3. Andrew Cumming, A gentle introduction to ML, Napier University, Edinburgh, 2013

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The content of the class is similar to the contents taught at other international universities. The students are encouraged to identify elements of functional programming in the current practice of IT companies running at the local level.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Understanding functional programming elements, Class participation, Homework		Midterm assessment, Writing exam		60
Applications		Quantity and quality of code in Lisp, Haskell and ML		Midterm assessment, Practical exam		40
10.4 Minimum standard of performance						
Understanding and code writing for the following concepts; Recursion, High Order Functions, Pattern Matching						

Course responsible
Lect.dr.eng. Octavian Pop

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	33.

2. Data about the subject

2.1	Subject name					Software Engineering					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof. dr eng. Eneia Todoran – Eneia.Todoran@cs.utcluj.ro					
2.4	Teachers in charge of applications					Assoc.prof. dr. Mitrea Paulina – Paulina.Mitrea@cs.utcluj.ro, Assoc.prof. dr. eng. Mitrea Delia Delia.Mitrea@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Software Engineering**	**2**	**-**	**1**	**1**	**28**	**-**	**14**	**14**	**74**	**130**	**5**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	25
Supplementary study in the library, online and in the field	17
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	17
Tutoring	5
Exams and tests	10
Other activities	0

3.7	Total hours of individual study	74
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. Pre-requisites (where appropriate)

4.1	Curriculum	Object Oriented Programming, Programming Techniques
4.2	Competence	Competences acquired in the above disciplines

5. Requirements (where appropriate)

5.1	For the course	Blackboard, projector, computer
5.2	For the applications	Computers, specific software

6. Specific competences

Professional competences	**C3 -** Problems solving using specific Computer Science and Computer Engineering tools (2 credits) **C3.1 -** Identifying classes of problems and solving methods that are specific to computing systems **C3.2 -** Using interdisciplinary knowledge, solution patterns and tools, making experiments and interpreting their results **C3.3 -** Applying solution patterns using specific engineering tools and mehods **C3.4 -** Comparatively and experimentaly evaluation of the alternative solutions for performance optimization **C3.5 -** Developing and implementing informatic solutions for concrete problems **C4 -** Improving the performances of the hardware, software and communication systems (1 credit) **C4.1 -** Identifying and describing the defining performance elements of hardware, software and communication systems **C4.2 -** Explaining the interaction of the factors that determine the performances of hardware, software and communication systems **C4.3 -** Applying fundamental methods and principles for increasing performance of hardware, software and communication systems **C4.4 -** Choosing criteria and methods for performance evaluation of hardware, software and communication systems **C4.5 -** Developing performance based professional solutions for hardware, software and communication systems **C5 -** Designing, managing the lifetime cycle, integrating and ensuring the integrity of hardware, software and communication systems (2 credits) **C5.1 -** Specifying the relevant criteria regarding the lifetime cycle, quality, security and the computing system's interaction with the environment and the human operator **C5.2 -** Using interdisciplinary knowledge for adapting an information system to application domain requirements **C5.3 -** Using fundamental principles and methods for security, reliability and usability assurance of computing systems **C5.4 -** Adequate utilization of quality, safety and security standards in information processing **C5.5 -** Creating a project including the problem's identification and analysis, its design and development, also proving an understanding of the basic quality requirements
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	The overall objective of discipline consists in the study and application of systematic, disciplined and quantifiable approaches in software systems development
7.2	Specific objectives	• Study and application of software development processes • Understanding the specific activities of software engineering • Knowledge of software engineering models • Knowledge of specific tools that can assist software engineers in the specification, design and validation process • Knowledge of methods for software modeling and performance analysis • Application of processes, methods and tools in small to medium-sized software projects

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction and overview of the course		
2	Software development paradigms: basic paradigms ('waterfall', prototyping, reusable components, formal methods), evolutionary paradigms (incremental development, spiral model, concurrent engineering)		
3	Modern processes: the unified process, agile methods and extreme		

	programming		
4	Basic activities (specification, development, validation, evolution): concepts, principles, processes		
5	Developing requirements: domain analysis, techniques for gathering requirements, capturing requirements as use cases		
6	Formal specification: formal modeling and analysis, model checking, tools in support of formal methods (PRISM)		
7	Modeling with classes: UML class and object diagrams, using design patterns		
8	Modeling with classes: the process of developing class diagrams, semantics of UML class diagrams, implementing class diagrams in Java		
9	Modeling interactions and behavior: UML interaction and state diagrams		
10	Modeling software behavior: UML state diagrams, software performance modeling and analysis.		
11	Architecting and designing software: design principles (increase cohesion, reduce coupling), architectural patterns (Layers, Pipe-and-Filter, etc.)		
12	Testing and inspecting to ensure high quality: testing techniques (equivalence partitioning, path testing) and integration strategies (top-down, bottom-up, scenario-based), inspections		
13	Use case driven development: use case specifications, analysis, design and implementation to realize the use cases, testing the use cases		
14	Program specifications: pre and post assertions, well-founded induction, declarative prototyping		

Bibliography

1. I. Sommerville. *Software Engineering* (6th, 7th, 8th, 9th, 10th editions). Addison Wesley (2001, 2004, 2006, 2010, 2015).
2. T. Lethbridge, R. Laganiere. *Object-Oriented Software Engineering: Practical Software Development using UML and Java* (2nd edition). McGraw-Hill, 2005. http://www.lloseng.com.
3. C. Baier, J.P. Katoen, *Principles of Model Checking*. MIT Press, 2008.
4. I. Nikolov. *Scala Design Patterns*. Packt Publishing 2016.
5. E.N. Todoran. *Inginerie software: studii in prototipizare si specificare formala*. Mediamira, Cluj-Napoca, 2006.

8.2. Applications (Laboratory)		Teaching methods	Notes
1	OCSF – an object client-server framework for reuse oriented development		
2	Simple Chat - an instant messaging system based on OCSF (1)		
3	Simple Chat - an instant messaging system based on OCSF (2)		
4	Using software modeling CASE tools: UML use case, class, interaction, state, component and deployment diagrams		
5	Using CASE tools for performance software modeling and analysis: PRISM, PEPA (1)		
6	Using CASE tools for performance software modeling and analysis: PRISM, PEPA (2)		
7	Test cases design with JUnit		
The **project class** attempts to simulate various aspects of the real world of software engineering. The students define the problem to be solved and the scope of the project under the supervision of the teaching assistant. Working alone is permitted, but they are encouraged to work in teams. The students must employ the paradigms and the software development methods that are presented in the taught course. They are expected to deliver three iterations of the project with predefined deadlines. For a traditional 'waterfall' project the deadlines correspond to requirements specification, design, and the final deliverable.			

Bibliography

1. T. Lethbridge, R. Laganiere. *Object-Oriented Software Engineering: Practical Software Development using UML and Java* (2nd edition). McGraw-Hill, 2005. http://www.lloseng.com.
6. PRISM manual, 2016. http://www.prismmodelchecker.org/manual/

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

Software Engineering is a well-established discipline in Computer Science and Information Technology. In this course, students acquire basic knowledge related to software development (paradigms, methods and tools) and learn to apply systematic and quantifiable approaches in the development of software systems. Course content has been developed based on interaction with specialists in Software Engineering from Romania,

Europe (UK, Greece) and Canada and has been rated by Romanian government agencies (CNEAA and ARACIS).

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Problem solving skills		Written examination		75%
Applications		Software design and validation skills		Laboratory colloquium, Project assessment		25%
10.4 Minimum standard of performance						
Development of a medium size software project using the skills taught in the Software Engineering course						

Course responsible
Prof.dr.eng. Eneia Todoran

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	34.

2. Data about the subject

2.1	Subject name					Introduction to Artificial Intelligence					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof. dr. eng. Leţia Ioan Alfred – Ioan.Alfred.Letia@cs.utcluj.ro					
2.4	Teachers in charge of applications					Assoc. prof. dr. eng. Groza Adrian – Adrian.Groza@cs.utcluj.ro Lect. dr. eng. Marginean Anca – Anca.Marginean@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Introduction to Artificial Intelligence**	**2**	**-**	**2**	**-**	**28**	**-**	**28**	**-**	**48**	**104**	**4**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28
Individual study								Hours
Manual, lecture material and notes, bibliography								18
Supplementary study in the library, online and in the field								5
Preparation for seminars/laboratory works, homework, reports, portfolios, essays								10
Tutoring								6
Exams and tests								9
Other activities								0

3.7	Total hours of individual study	48
3.8	Total hours per semester	104
3.9	Number of credit points	4

4. Pre-requisites (where appropriate)

4.1	Curriculum	Logic Programming, Functional Programming
4.2	Competence	Elementary fundamentals of programming

5. Requirements (where appropriate)

5.1	For the course	Projector, Computer
5.2	For the applications	Computers with Linux, Specific Software

6. Specific competences

Professional competences	**C3 –** Problems solving using specific Computer Science and Computer Engineering tools (1 credit) **C3.1 –** Identification of classes of problems and the methods to solve them characteristic of information systems **C3.2 –** Usage of interdisciplinary knowledge, patterns of solutions and tools, experimentation and interpretation of their results **C3.3 –** Aplication of solution patterns using engineering tools and methods **C3.4 –** Comparative evaluation, including experiments, of alternative solutions, to optimize performance **C3.5 –** Development and implementation of computational solutions for concrete problems **C5 –** Designing, managing the lifetime cycle, integrating and ensuring the integrity of hardware, software and communication systems (1 credit) **C5.1 –** Stating the criteria relevant to quality, security and system interaction with the environment and human operator **C5.2 –** Usage of interdisciplinary knowledge for the adaptation of the informatic system to the requirements of the application domain **C5.3** – Using fundamental principles and methods for ensuring the security, the safety and ease of exploitation of the computing systems **C5.4 –** Adequate utilization of quality, safety and security standards in information processing **C5.5 –** Realization of a project including problem identification and analysis, design and development, while proving the understanding of the basic quality needs and requirements **C6 –** Designing intelligent systems (2 credits) **C6.1 –** Describing the intelligent systems' components **C6.2 –** Using domain-specific tools for explaining the operation of intelligent systems **C6.3 –** Applying the main methods and principles for specifying solutions for typical problems using intelligent systems **C6.4 –** Choosing criteria and methods for the evaluation of quality, performances and limitations of information systems **C6.5 –** Developing and implementing professional projects for intelligent systems
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Knowledge of representation and reasoning of fundamental problems of artificial intelligence
7.2	Specific objectives	Fundamental search methods, Usage of first-order logic and description logics, Basic planning representation and solving methods

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction.	Slides, Algorithms, Quality of solutions, Exceptions, Limits in the representation of the real world,	
2	Intelligent Agents: behavior, environments, structure .		
3	Solving Problems by Searching: uninformed, searching with partial information.		
4	Informed Search Methods and Exploration: heuristics, local search algorithms and optimization problems, local search in continuous spaces.		
5	Constraint Satisfaction Problems: backtracking, local search.		
6	Adversarial Search: alpha-beta pruning, imperfect, real-time decisions, games that include an element of chance.		
7	Logical Agents: knowledge-based agents, propositional logic, effective propositional inference.		
8	First-Order Logic.		
9	Inference in First-Order Logic: forward, backward chaining, resolution.		
10	Knowledge Representation.		
11	Description logics: description languages, terminologies, world description, inferences, reasoning algorithms, language extensions		

12	Planning: partial-order planning, planning graphs.		
13	Planning and Acting in the Real World: schedules and resources, hierarchical network planning, conditional planning, execution monitoring and re-planning, continuous planning.		
14	Course Overview.		
Bibliography 1. Artificial Intelligence: A Modern Approach: Russell, Norvig, Prentice Hall, 2002 2. Basic Description Logics: Baader, Nutt, CUP, 2003			
8.2. Applications (Laboratory)		Teaching methods	Notes
1	Introduction to the documentation for the assignment	Platform, Documentation, Testing, Examples, New examples	
2	Studying the documentation for the assignment		
3	Studying the design of the tool		
4	Practicing the exercises provided in the archive		
5	Understanding the main parts of the software		
6	Running the system by tracing at high level		
7	Mastering the running of the system and the examples provided		
8	Conceptual design of new examples		
9	Code for the new examples		
10	Testing and debugging the new cases		
11	Measuring the performance of the system		
12	Documenting the new scenarios		
13	Comparison of the differences between the cases developed and those provided		
14	Final evaluation of the exercises developed		
Bibliography 1. Various Artificial Intelligence Tools from the WWW			

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The textbook is one of the most known and used one in the world of the best universities, continuously assessed by the university and research community in the world regarding its influence and use in the software oriented companies.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Problems and theoretical concepts		Written exam		80%
Applications		Usage of specific tools on the examples developed and tested by the students		Evaluation in the laboratory		20%
10.4 Minimum standard of performance						
Representation of knowledge and its use in solving specific problems using specific tools						

Course responsible
Prof.dr.eng. Ioan Alfred Letia

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	35.

2. Data about the subject

2.1	Subject name					Economic law					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Assoc.prof.dr.jur. Roxana Cordos – Roxana.Cordos@mis.utcluj.ro					
2.4	Teachers in charge of applications					-					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	Colloquium	2.8	Subject category	DC/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Economic legislation**	**2**	**-**	**-**	**-**	**28**	**-**	**-**	**-**	**24**	**52**	**2**

3.1	Number of hours per week	2	3.2	of which, course	2	3.3	applications	-
3.4	Total hours in the teaching plan	28	3.5	of which, course	28	3.6	applications	-

Individual study	Hours
Manual, lecture material and notes, bibliography	18
Supplementary study in the library, online and in the field	2
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	
Tutoring	2
Exams and tests	2
Other activities	

3.7	Total hours of individual study	24
3.8	Total hours per semester	52
3.9	Number of credit points	2

4. Pre-requisites (where appropriate)

4.1	Curriculum	Not the case
4.2	Competence	Not the case

5. Requirements (where appropriate)

5.1	For the course	Not the case
5.2	For the applications	Not the case

6. Specific competences

Professional competences	N/A

Cross competences	**CT3 –** Demonstrating the spirit of initiative and action for updating professional, economical and organizational culture knowledge (2 credits)

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Applying the general and specific knowledge of technical culture in solving the business issues in this field
7.2	Specific objectives	Knowing the basic legislation in the field and finding solution for different types of problems.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	General notions of economic legislation.	Modern teaching methods	
2	The merchants. Commerce acts		
3	Bank operations.		
4	Commercial contracts –general notions		
5	Classification of contracts		
6	The contract of sale		
7	The contract leasing.		
8	General rules applied to commercial societies		
9	The constitutive act of a firm		
10	Changes in the constitutive act of a firm.		
11	Types of commercial societies.		
12	The insolvency procedure.		
13	The working contract		
14	General notions of economic legislation.		

Bibliography

1. Bacali, L (coord), Antreprenoriat-manualul calificarii, UTPress, 2010 (biblioteca UTCN)
2. Bodu S., Drept commercial completat cu notiuni fundamentale de drept civil- curs universitar, 2005 (biblioteca UTCN)
3. S.Angheni, M.Volonciu, C.Stoica, M.Lostun, Drept comercial, Ed. Oscar Print, Bucuresti, 2000
4. S.Carpenaru, Drept comercial, Ed.All, Bucuresti, 2007
5. I.L.Georgescu, I.Bacanu, Drept comercial român, vol.II, Ed.Lumina Lex, Bucuresti, 2000

8.2. Applications (Seminars, Laboratory, Projects)		Teaching methods	Notes
1	-		

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The students will have the possibility to learn how to put into practice a business idea in the studied domain.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Colloquium		Written test		100%
Applications						
10.4 Minimum standard of performance						
Grade 5						

Course responsible
Assoc.prof.dr.jur. Roxana Cordos

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	36.

2. Data about the subject

2.1	Subject name					Graphical Processing Systems					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof.dr.eng. Gorgan Dorian – dorian.gorgan@cs.utcluj.ro					
2.4	Teachers in charge of applications					Lect.dr.eng. Bacu Victor, As.eng. Constantin Nandra, {victor.bacu, constantin.nandra}@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Graphical Processing Systems**	**2**	**-**	**2**	**-**	**28**	**-**	**28**	**-**	**48**	**104**	**4**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	20
Supplementary study in the library, online and in the field	6
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	10
Tutoring	3
Exams and tests	9
Other activities	0

3.7	Total hours of individual study	48
3.8	Total hours per semester	104
3.9	Number of credit points	4

4. Pre-requisites (where appropriate)

4.1	Curriculum	Computer programming (C language) Elements of Computer Assisted Graphics
4.2	Competence	Applications development in C programming language, Graphical systems architecture, The graphical processing pipeline

5. Requirements (where appropriate)

5.1	For the course	Projector, computer
5.2	For the applications	Laboratory attendance is mandatory Study of laboratory materials from the server

6. Specific competences

Professional competences	**C4 –** Improving the performances of the hardware, software and communication systems (4 credits) **C4.1 –** Identifying and describing the defining elements of the performances of the hardware, software and communication systems **C4.2 –** Explaining the interaction of the factors that determine the performances of the hardware, software and communication systems **C4.3 –** Applying the fundamental methods and principles for increasing the performances of the hardware, software and communication systems **C4.4 –** Choosing the criteria and evaluation methods of the performances of the hardware, software and communication systems **C4.5 –** Developing professional solutions for hardware, software and communication systems based on performance optimization
Cross competences	N/A

7. Discipline objectives (as results from the key competences gained)

7.1	General objective	Study and experiment with the 3D photorealistic algorithms. Development of 2D and 3D graphics applications.
7.2	Specific objectives	1. Creation of the graphic model of a 3D scene of objects 2. Implementation and usage of the fundamental 3D graphics algorithms that can be found in the core of a graphic system 3. Development of graphic applications in a high-level programming language (C, C++) based on graphics libraries (ex. OpenGL) 4. Implementation of the main phases of the graphics transformation pipeline, in order to transform a 3D scene into an image.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Computational graphics	New multimedia teaching approaches will be used in classes. The course is interactive and includes demonstrations that exemplify graphical methods and algorithms.	During the semester and before each exam there are a few preparation hours planned.
2	Hidden line and surface removal algorithms. Part 1		
3	Hidden line and surface removal algorithms. Part 2		
4	3D objects modeling		
5	Particles based models		
6	Polygonal objects rendering. Part 1		
7	Polygonal objects rendering. Part 2		
8	Illumination models. Local reflection model. Phong model		
9	Shadow computation		
10	Texture mapping. Part1		
11	Texture mapping. Part2		
12	Global reflection models. Ray-tracing algorithm		
13	Global reflection models. Radiosity algorithm		
14	Graphical animation		

Bibliography

8. Watt A., "*3D Computer Graphics*". Addison-Wesley, 1998.
9. Watt A., Policarpo F.:"*3D Games. Real-time Rendering and Software Technology*". Addison-Wesley, 2001.
10. Akenine-Moller T., Haines E., "*Real-Time Rendering"*. A.K. Peters 2nd edition, 2002.
11. Foley J.D., van Dam, A., Feiner, S.K., Hughes, J.F., "*Computer Graphics. Principles and Practice*". Addison-Wesley Pblishing Comp., 1992.
12. Gorgan D., Rusu, D., "*Elemente de Grafică pe Calculator*". Cluj-Napoca, 1996.

In virtual library

1. Course resources, http://cgis.utcluj.ro/teaching/

8.2. Applications (Laboratory)		Teaching methods	Notes
1	Mathematics for computer graphics	Documentation and examples will be available	Each student will have to develop a
2	Introduction to modern OpenGL		
3	Basic vertex and fragment shaders		

4	Debugging methods	to the students, prior to the laboratory classes, on a dedicated server. The students will work independently but will also be assisted by the teacher.	specific project based on the knowledge acquired at the laboratory hours.
5	3D Transformations		
6	3D models and textures		
7	First project evaluation		
8	Lighting model - Part 1		
9	Lighting model - Part 2		
10	Shadow mapping		
11	Second project evaluation		
12	Cube maps and environmental mapping		
13	Normal mapping		
14	Final project assessment		

Bibliography

In virtual library

1. Curse and practical works, http://cgis.utcluj.ro/teaching/

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

This discipline is integrated into the Computers and Information Technology domain. The content is classic, yet modern, and introduces to students the fundamentals of 3D graphic systems and algorithms. The content of this discipline has been aligned with the information presented in similar disciplines from other major universities and companies from Romania, Europe and USA and has been evaluated by the authorized Romanian governmental agencies (CNEAA and ARACIS).

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		The written exam evaluates the understanding of the information presented in classes and the ability to apply this knowledge.		Evaluation is performed through written exam.		60% (E)
Course activity		The activity in class evaluates the active involvement of the students in the teaching process and their participation to the discussions, debates and other class activities during the entire semester.		Evaluation is performed through a very short tests.		10% (AC)
Applications		Laboratory assessment evaluates the practical abilities obtained by the students. Through homework assignments the students have the opportunity to develop their skill in applying the notions, concepts and methods presented in class.		Evaluation is performed through written and practical exam.		40% (L)
10.4 Minimum standard of performance						
Graduation requirement: M≥5, final mark M=0.5*E+0.4*L+0.1*AC						

Course responsible
Prof.dr.eng. Dorian Gorgan

Head of department
Prof.dr.eng. Rodica Potolea