SYLLABUS

1. Data about the program of study

1.1 Institution	The Technical University of Cluj-Napoca
1.2 Faculty	Faculty of Automation and Computer Science
1.3 Department	Computer Science
1.4 Field of study	Computer Science and Information Technology
1.5 Cycle of study	Master
1.6 Program of study / Qualification	Artificial Intelligence and Vision
1.7 Form of education	Full time

2. Data about the subject

2.1 Subject name	***Programming Engineering***			Subject code	**14.00**
2.2 Course responsible / lecturer	Prof.dr.ing. Dînșoreanu Mihaela - mihaela.dinsoreanu@cs.utcluj.ro				
2.3 Teachers in charge of seminars / Laboratory / project	Prof.dr.ing. Dînșoreanu Mihaela - mihaela.dinsoreanu@cs.utcluj.ro				
2.4 Year of study	II	2.5 Semester	1	2.6 Type of assessment (E - exam, C - colloquium, V – verification)	E
2.7 Subject category	Formative category: DA – advanced, DS – speciality, DC – complementary				DA
	Optionality: DI – imposed, DO – optional (alternative), DF – optional (free choice)				DI

3. Estimated total time

3.1 Number of hours per week	3	of which:	Course	2	Seminars	-	Laboratory	1	Project	-
3.2 Number of hours per semester	42	of which:	Course	28	Seminars	-	Laboratory	14	Project	-

3.3 Individual study:	
(a) Manual, lecture material and notes, bibliography	20
(b) Supplementary study in the library, online and in the field	15
(c) Preparation for seminars/laboratory works, homework, reports, portfolios, essays	15
(d) Tutoring	5
(e) Exams and tests	3
(f) Other activities:	-

3.4 Total hours of individual study (suma (3.3(a)...3.3(f)))	58
3.5 Total hours per semester (3.2+3.4)	100
3.6 Number of credit points	4

4. Pre-requisites (where appropriate)

4.1 Curriculum	Software design
4.2 Competence	Development of techniques, technologies, methods and methodologies specific for information systems: Innovative design of specific information systems

5. Requirements (where appropriate)

5.1. For the course	Projector, board, computer with internet access
5.2. For the applications	Projector, computers with internet access

6. Specific competence

6.1 Professional competences	• analyse software specifications • analyse software architectures • create data models • create software solutions • define technical requirements • design process • use data processing techniques • acquire system component • align software with system architectures • manage system testing • assess ICT knowledge • manage database • implement ICT security policies • integrate system components • design enterprise architecture • design information system • use an application-specific interface • use markup languages • build predictive models • build recommender systems • utilise machine learning • conduct scholarly research • develop computer vision system • apply statistical analysis techniques • manage data collection systems
6.2 Cross competences	The graduate: • develops an analytical approach • takes a proactive approach • develops strategies to solve problems • is open minded • coordinates engineering teams

7. Expected Learning Outcomes

Knowledge	The student has knowledge of: • algorithms • artificial neural networks • business process modelling • computer programming (Python) • data mining • data models • data science • information architecture • information extraction • information structure • principles of artificial intelligence • query languages • systems development life cycle • task algorithmisation • unstructured data • visual presentation techniques • machine learning (computer programming) • business analytics • business intelligence • computer vision • database development tools • deep learning • operational research • database development tools • system design • systems development life cycle • systems theory • web programming • ICT process quality models • ICT project management methodologies • ICT security legislation • ICT system integration • ICT system programming • computer programming • defence standard procedures • information structure • model-based systems engineering • security engineering

Skills	The student is able to: • analyse big data • analyse business requirements • apply ICT systems theory • create data sets • define technical requirements • design processes • develop statistical software • use data processing techniques • apply systemic design thinking • assess ICT knowledge • build predictive models • build recommender systems • design application interfaces • design database schemas • identify processes for re-engineering • manage ICT data classification • manage ICT semantic integration • manage business knowledge • perform dimensionality reduction • utilise machine learning • acquire system components • align software with system architectures • apply ICT systems theory • create data models • design enterprise architecture • design information systems • implement ICT security policies • integrate system components • manage databases • manage system testing • use an application-specific interface
Responsibilities and autonomy	The student has the ability to work independently in order to: • develop an analytical approach • take a proactive approach • develop strategies to solve problems • be open-minded • coordinate engineering teams

8. Discipline objective (as results from the *key competences gained*)

8.1 General objective	The main objective of this discipline is to provide specific information and prepare students to create software projects that meet quality requirements. Thus, it aims to provide the ability to analyze different architectural and design alternatives, to make the most appropriate architectural decisions for the context in order to design any type of application, with an emphasis on optimizing its performance as much as possible.
8.2 Specific objectives	To achieve these general objectives, students will: • Aim to understand and solve both the functional requirements of a software system and its quality requirements (availability, performance, security, scalability, etc.) • Study existing architectural solutions at different levels of granularity (architectural styles, architectural and design patterns) • Study evaluation metrics of different qualitative aspects (complexity, reliability, availability, etc.) • Learn to analyze requirements and design applicable architectural

	alternatives; • Learn to evaluate architectural solutions by applying specific evaluation models;

9. Contents

9.1 Lectures	Hours	Teaching methods	Notes
Introduction	2	Blackboard presentation, video projector presentation, discussions, course materials, quizzes, Moodle platform	
Fundamentals of Metrics and Measurement	2		
Agile Process Models and Practices	2		
Requirements Engineering	2		
Planning (Measuring Size, Complexity, Effort)	2		
Planning (Estimating Time and Cost)	2		
Monitoring (Metrics)	2		
Defect Metrics	2		
Reliability Metrics	2		
Availability Metrics	2		
Scalability Metrics	2		
Security	2		
Architecture Evaluation Models	2		
Recap and Conclusions	2		

Bibliography:

- Software Metrics: A Rigorous and Practical Approach, by Norman Fenton, James Bieman, Third Edition, CRC Press, Inc., Boca Raton, FL, 2014
- Software Engineering, 10th Edition, Ian Sommerville, Pearson ed. 2016
- Righting software, Juval Lowy, O'Reilly, 2020
- Software Architecture: The Hard Parts, Ford, Neal, Richards, Mark, Sadalage, Pramod, & Dehghani, Zhamak. (2021). O'Reilly Media, Inc.
- Diferite articole

9.2 Applications - Seminars/Laboratory/Project	Hours	Teaching methods	Notes
Complexity Measurement	2	Presentation, Discussions	
Effort Estimation	2		
Project Planning	2		
Defect Metrics	2		
Reliability Metrics	2		
Availability Metrics	2		
Scalability Metrics	2		

Bibliography

- Software Metrics: A Rigorous and Practical Approach, by Norman Fenton, James Bieman, Third Edition, CRC Press, Inc., Boca Raton, FL, 2014
- Software Engineering, 10th Edition, Ian Sommerville, Pearson ed. 2016
- Righting software, Juval Lowy, O'Reilly, 2020
- Software Architecture: The Hard Parts, Ford, Neal, Richards, Mark, Sadalage, Pramod, & Dehghani, Zhamak. (2021). O'Reilly Media, Inc.
- Diferite articole

**Se vor preciza, după caz: tematica seminariilor, lucrările de laborator, tematica și etapele proiectului.*

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

Since this discipline is very important for the qualitative evaluation of software system design, its content is as modern as possible because it recapitulates the principles of software design, the principles of project design, then metrics for evaluating software projects from different quality perspectives. The content of the discipline was discussed with important actors in this field, both academic and industrial, from Romania, Europe and the U.S. The discipline was evaluated, together with the master's study program Artificial Intelligence and Vision, by ARACIS.

10. Evaluation

Activity type	Assessment criteria	Assessment methods	Weight in the final grade
Course	Ability to solve problems specific to the field Presence, (inter)activity during classes	Quizes during classes (continuous), Written Exam - summative	60%
Seminar	Ability to solve problems specific to the field Presence, (inter)activity during classes	Presentation of a relevant research topic - summative	40%
Laboratory	-	-	-
Project	-	-	-
Minimum standard of performance: Seminar grade >=5; Exam grade >=5			

Date of filling in:	**Responsible**	**Title First name Last name**	**Signature**
	Course	Prof.dr.ing. Mihaela DÎNȘOREANU	
	Applications	Prof.dr.ing. Mihaela DÎNȘOREANU	

Date of approval in the department	Head of department, Prof.dr.eng. Rodica Potolea
Date of approval in the Faculty Council	Dean, Prof.dr.eng. Vlad Mureșan