University of Naples "Federico II"

MASTER'S DEGREE IN COMPUTER ENGINEERING

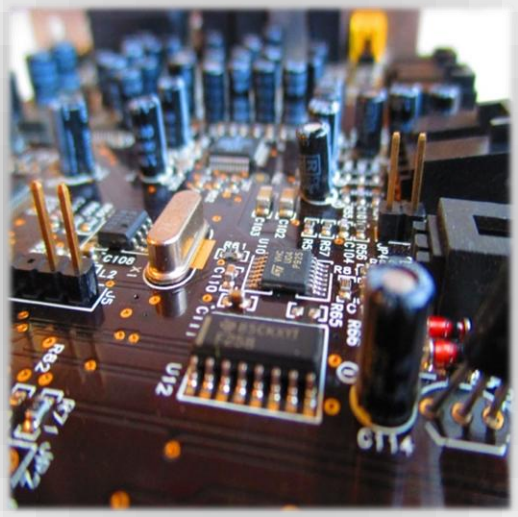
Scuola Politecnica e delle Scienze di Base

Department of Electrical Engineering and Information Technologies

Class of Master's Degrees in Computer Engineering LM-32

Curricular paths

The professional training of the graduate in Computer Engineering the acquisition of advanced design skills and innovative contents in the area of **Internet and Cloud**, **Embedded Systems for Industry and IoT**, the area of **Data Engineering and Artificial Intelligence**, in the area of **Cyber-Security** and in that of **Intelligent Robotic Systems.**



The **Internet and Cloud** area aims to provide students with skills in advanced networking Technologies, in order to train professionals capable of handling the complex set of protocols, architectures and applications in use currently and in the future (with reference to cloud systems, to modern wireless networks, to the infrastructure-as-code paradigm). In this context the principles for network design and management are of particular interest, they analyse the issues related to resource management, traffic routing and robustness to failures and attacks, both in traditional networks and in advanced networks.

The area **Embedded Systems for Industry and IoT** focuses on methodologies and technologies for the design and implementation of so-called "dedicated" processing systems, whose diffusion is constantly increasing and which represent the heart of the modern applications of Industry 4.0 and of numerous other sectors and systems (vehicle on-board systems, telemedicine, multimedia devices, robotic systems, Internet of Things (IoT), systems for the supervision and control of infrastructure, etc.). In this context, the performance requirements, the scalability, the reliability, the real-time processing, and the advanced architectures for high-performance computing are very important.

Curricular paths

The area **Data Engineering and Artificial Intelligence** aims to train engineers experts in the design and maintenance of complex computer systems able to manage efficiently, through appropriate hardware/ software infrastructure, large amounts of data and extract "useful" knowledge from them, through the most advanced "data-centric" analytics techniques. At the same time, tools that have become indispensable in the field of modern IT systems are provided to support the operational and management processes of any organization. The issues related to the implementation of "intelligent" systems, based on the latest methodologies and techniques of Artificial Intelligence (AI), from machine/deep learning to cognitive systems, are then addressed in detail, from information retrieval to machine vision, as well as those related to their production and testing in accordance with the MLOps paradigm and aspects related to the ethics of AI models in different application contexts.

The **Cyber-Security** area aims to train new "security engineers", that is, professionals competent in the protection of systems, software and networks. In this field, which now has a strategic importance both nationally and internationally, some of the most requested profiles are the following: the Security Administrator - which has the task of making operational the technological security solutions; the Security Architect - who thanks to his modelling skills, draws the security measures and policies adopted by the organization; the Security Engineer - who has a technical and modelling baggage, is responsible for monitoring systems and proposing solutions related to incident response; the Security Analyst - which has process analysis skills and is in charge of assessing vulnerabilities that may affect networks, equipment, applications and services proposing solutions and practical solutions. The subject is therefore complex and requires highly qualified professionals with specialist skills in systems engineering, software engineering and communication networks.

Curricular paths

The area "**Intelligent Robotic Systems**" provides Computer Engineering students with the fundamental methodological skills necessary for the modeling, planning, and control of robotic systems, as well as the hardware and software tools for developing intelligent robotic applications. The course will delve into topics related to real-time operating systems, the design of systems for the acquisition and processing of sensory information (e.g., implementation of drivers for interfacing with devices), the configuration and use of machine learning algorithms for knowledge extraction and representation, and for autonomous robot action planning. A computer engineer trained in this area will be able to design and implement intelligent robotic systems capable of supporting complex cognitive tasks, with reference to sensor systems, control systems, and decision support algorithms, effectively interacting with experts in various application fields.

Course overview

The aim of the Master's Degree in Computer Engineering is to train a professional able to fit into highly differentiated and rapidly changing production realities, in the fields of design, engineering, development, maintenance of services, applications and computer systems, of complex business information systems.
The professional training of the graduate in Computer Engineering requires the acquisition of advanced design skills and innovative content in the area of computer systems.
The master's degree programme in Computer Engineering is organized in the following thematic areas.
A first thematic area of educational interest is **Data Engineering and Artificial Intelligence**. The goal of this subject area is the training of engineers experienced in designing complex computer systems able to efficiently manage large amounts of data and extract "useful" knowledge from the latter through advanced analytics techniques.
Another thematic area is related to **Embedded Systems for Industry and IoT.** Embedded systems are information processing systems designed to run specific applications. Their diffusion is constantly increasing, they represent the heart of modern industrial applications and are present in many fields of application: automotive systems, telemedicine, multimedia devices, robotic systems, home automation, Internet of Things , devices for security management, real time systems, systems for infrastructure control and computing systems for AI applications.
Internet and cloud is a thematic area that aims to provide students with skills on advanced technologies in the field of networking in order to train professionals able to handle the complex set of protocols, architectures and applications currently in use and in the future.
An additional thematic area of interest for the educational path of the master's degree in Computer Engineering is **Intelligent Robotic Systems**, related to intelligent robotics. This is an emerging technology that is rapidly growing, with a high potential for innovation and a variety of applications in industrial, domestic, and service sectors, as well as in logistics, healthcare, surveillance systems, and more.
Finally, the theme of **Cyber-Security** is of strategic importance in the world of Information Technology, both nationally and internationally. The aim of this thematic area is to train the new "security engineers", that is, professionals competent in the protection of systems, software and networks.

Career opportunities

With a growing demand for engineering/IT skills, the specific preparation of the graduate in Computer Engineering makes him one of the most sought-after professionals in the job market.
The majority of graduates in Computer Engineering are absorbed by the numerous service companies and industries of the national and international territory.
Employment opportunities include the following:
- the public and private services sector;
- consultancy firms;
- undertakings producing and supplying computer and industrial equipment and systems;
- companies developing products and services with a high IT content;
- public administrations;
- industry in general.

Admission procedure

The required curricular requirements include the achievement of a BSc in class L-8. In addition, access is subject to verification of the curricular requirements and the adequacy of the student's preparation. Further information can be found at the following link: LM informatica

Curricular internship activities

The course of study consists of 3 CFU of "**Other Training Activities**" and 9 CFU of "**Stage or internships in enterprises**" that can be acquired through extramoenia or intramoenia internships.
The internship is carried out in companies, research centers or other public and/ or private, Italian or foreign, entities with the support of a tutor of the company and the supervision of a university tutor. The intramoenia internship is carried out at the university's research laboratories with the support of a university tutor (teacher or researcher).

Final exam

The Master's Degree in Computer Engineering is obtained after taking a final test at which you are admitted once you have completed all the credits provided for in your study plan, excluding those relating to the final test itself.
The teaching committee of the course of study has set up an automatic system to support students in the choice, management and final drafting of the final exam (all information can be found at the following link: M63 degree).
The final exam consists in the evaluation of the Master's Degree Thesis, a written report (possibly in English) on an original project independently developed by the student under the guidance of a supervisor and possible assistant supervisor. The document presenting the guidelines for the computation of final scores is published on the website of the degree programme.
The thesis focuses on training activities carried out within one or more courses or internship activities. The candidate is allowed to use an audio-visual medium, to be projected publicly, or, alternatively, to write a summary file, to be handed in a copy to each member of the Commission.
At the end of the presentation, each teacher can address observations to the candidate, related to the topic of thesis work. The presentation shall normally last 10 minutes.

Minimum curricular requirements for admission to the Master's Degree in Computer Engineering (LM-32)

The student holding the title of Degree ex D.M. 509/99 or ex D.M. 270/04 may be admitted to the Master of Science in Computer Engineering if he/she has acquired in the previous career a number of CFU in the following disciplinary scientific fields to the minimum extent correspondingly indicated:

SSD	CFU minimum
MAT/05, MAT/03, MAT/09, FIS/01, FIS/03, MAT/08	24
ING-INF/01, ING -INF/02, ING -INF/03, ING -INF/04, ING-INF/05, ING -INF/06, ING -INF/07, ING -IND/13, ING-IND/16, ING -IND/17, ING -IND/31, ING -IND/32, ING-IND/34, ING-IND/35, INF/01	54

Contacts

Didactic Coordinator of the Study Programmes in Computer Engineering:
Prof. Simon Pietro Romano
Department of Electrical Engineering and Information Technologies
Tel. 081-7683823 – email: spromano@unina.it

Contact person of the Degree Course for the ERASMUS Program:
Prof. Alessio Botta
Department of Electrical Engineering and Information Technologies – Tel. 081/7683823 – Tel. 081-7683865– email: alessio.botta@unina.it

Contact person of the Degree Course for internships:
Prof. Antonio Montieri
Department of Electrical Engineering and Information Technologies Tel. 081/7683856 – email: antonio.montieri@unina.it

Contact person for Orientation:
Prof. Michela Gravina
Department of Electrical Engineering and Information Technologies
Email: michela.gravina@unina.it

Study plan

Visit the following link to see all the information on the study plan, the assignments of the chairs and further information.

First year courses

Teaching name	Sem.	CFU	SSD	Subject Area	TEA(*)
Architettura dei Sistemi Digitali	I	6	ING-INF/05	Ingegneria Informatica	B
Algorithms, Data Structures and Machine Learning: *Modulo di Algorithms and Data Structures*	I	6	ING-INF/05	Ingegneria Informatica	B
Related or supplementary education activities chosen by the student (the student will have to choose from 0 to 2 courses in **Table A**, note a)	I	0-18		Attività formative affini o integrative	C
Educational activities chosen by the student (**Table C**, note b)	I	0-18		A scelta dello studente	D
Architettura e Progetto dei Calcolatori	II	9	ING-INF/05	Ingegneria Informatica	B
Networks and Cloud Infrastructures	II	6	ING-INF/05	Ingegneria Informatica	B
Algorithms, Data Structures and Machine Learning: *Modulo di Machine Learning*	II	6	ING-INF/05	Ingegneria Informatica	B
Related or supplementary education activities chosen by the student (the student will have to choose from 0 to 2 courses in **Table A**, note a)	II	0-18		Attività formative affini o integrative	C
Educational activities chosen by the student (**Table C**, note b)	II	0-18		A scelta dello studente	D

Note a) - The student can choose up to **18 type C CFU** from Table A, to be attended in the first semester and/or in the second semester of the **first year**.
Note b) - The student can choose up to **18 type D CFU** from Table C, to be attended in the first and/or second semester of the **first and/or second year**.

Second year courses

Teaching name	Sem.	CFU	SSD	Subject Area	TEA(*)
Impianti di elaborazione	I	9	ING-INF/05	Ingegneria Informatica	B
Curricular training activities chosen by the student (The student must choose at most 3 courses from one of the areas in **Table B**)	I	0-18	ING-INF/05	Ingegneria Informatica	B
Educational activities chosen by the student (**Table C**, note b)	I	0-18		A scelta dello studente	D
Curricular training activities chosen by the student (The student must choose at most 3 courses from one of the areas in **Table B**)	II	0-18	ING-INF/05	Ingegneria Informatica	B
Educational activities chosen by the student (**Table C**, note b)	II	0-18		A scelta dello studente	D
Other training activities	II	3		Ulteriori attività formative	F
Laboratory activities or internships in companies	II	9		Stage o tirocini	F
Final exam	II	12		Ulteriori attività formative	E

Note b) - The student can choose up to **18 type D CFU** from Table C, to be attended in the first and/or second semester of the **first and/or second year**.

Table A

Related or supplementary education activities chosen by the student

Teaching or training activities	Sem.	CFU	SSD	Subject Area	TEA(*)	Mutations
Trasmissione dei segnali digitali	I	9	ING-INF/03	Attività formative affini o integrative	C	
Teoria dell'Informazione	I	9	ING-INF/03	Attività formative affini o integrative	C	LM-TLMD
Architetture dei sistemi integrati	I	9	ING-INF/01	Attività formative affini o integrative	C	LM-IELM
Economia ed Organizzazione Aziendale	I	9	ING-IND/35	Attività formative affini o integrative	C	
Algoritmi di Ottimizzazione Combinatoria e su Rete	II	9	MAT/09	Attività formative affini o integrative	C	
Calcolo Scientifico per l'Innovazione Tecnologica	II	9	MAT/08	Attività formative affini o integrative	C	
Elaborazione dei segnali multimediali	II	9	ING-INF/03	Attività formative affini o integrative	C	LM-TLMD

*The student can choose to use the **18 CFU** of the table above, in full in the first semester and/or in the second semester of the first year.*

Table B

Curricular training activities chosen by the student

Area	Teaching or training activities	CFU	SSD	Sem.	Subject Area	TEA(*)
Data Engineering and Artificial Intelligence	Information Systems and Business Intelligence	6	ING-INF/05	I	Ingegneria Informatica	B
	AI Systems Engineering	6	ING-INF/05	I	Ingegneria Informatica	B
	Information Retrieval	6	ING-INF/05	II	Ingegneria Informatica	B
	Big Data Engineering	6	ING-INF/05	II	Ingegneria Informatica	B
Embedded Systems for Industry and IoT	Distributed Systems and IoT	6	ING-INF/05	I	Ingegneria Informatica	B
	High Performance and Quantum Computing	6	ING-INF/05	I	Ingegneria Informatica	B
	Real Time Systems and Industrial Applications	6	ING-INF/05	II	Ingegneria Informatica	B
	Embedded Systems	6	ING-INF/05	II	Ingegneria Informatica	B
Internet and Cloud	Web and Real Time Communication Systems	6	ING-INF/05	I	Ingegneria Informatica	B
	Decentralized Applications and Blockchain	6	ING-INF/05	I	Ingegneria Informatica	B
	Wireless Networks and IoT Technologies	6	ING-INF/05	II	Ingegneria Informatica	B
	Cloud Platforms and Infrastructure-as-Code	6	ING-INF/05	II	Ingegneria Informatica	B
Cyber-Security	System Security	6	ING-INF/05	I	Ingegneria Informatica	B
	Network Security	6	ING-INF/05	I	Ingegneria Informatica	B
	Data Analysis and Cybersecurity	6	ING-INF/05	II	Ingegneria Informatica	B
	Software Security	6	ING-INF/05	II	Ingegneria Informatica	B
Intelligent Robotic Systems (note c)	Robotics: modulo di Robotic Systems	6	ING-INF/04	I	Ingegneria dell'automazione e robotica	B
	Robotics: modulo di Robotic Lab	6	ING-INF/04	II	Ingegneria dell'automazione e robotica	B

	Un insegnamento a scelta tra:					
	Information Systems and Business Intelligence	6	ING-INF/05	I	Ingegneria Informatica	B
	AI Systems Engineering	6	ING-INF/05	I	Ingegneria Informatica	B
	Distributed Systems and IoT	6	ING-INF/05	I	Ingegneria Informatica	B
	High Performance and Quantum Computing	6	ING-INF/05	I	Ingegneria Informatica	B
	Web and Real Time Communication Systems	6	ING-INF/05	I	Ingegneria Informatica	B
	Decentralized Applications and Blockchain	6	ING-INF/05	I	Ingegneria Informatica	B
	System Security	6	ING-INF/05	I	Ingegneria Informatica	B
	Network Security	6	ING-INF/05	I	Ingegneria Informatica	B
	Big Data Engineering	6	ING-INF/05	II	Ingegneria Informatica	B
	Information Retrieval	6	ING-INF/05	II	Ingegneria Informatica	B
	Real Time Systems and Industrial Applications	6	ING-INF/05	II	Ingegneria Informatica	B
	Embedded Systems	6	ING-INF/05	II	Ingegneria Informatica	B
	Wireless Networks and IoT Technologies	6	ING-INF/05	II	Ingegneria Informatica	B
	Cloud Platforms and Infrastructure as-a-Code	6	ING-INF/05	II	Ingegneria Informatica	B
	Data Analysis and Cybersecurity	6	ING-INF/05	II	Ingegneria Informatica	B
	Software Security	6	ING-INF/05	II	Ingegneria Informatica	B

*The student has to choose 3 courses from one of the areas in **Table B***
***Note c)** in case the student chooses the "intelligent Robotic Systems" area, it will mandatory to select the "Robotics" course (the course includes a "Robotic Systems" module during the I semester and a "Robotic Lab" course during the II semester)*

Table C

Other educational activities chosen by the student

Teaching activity	CFU	SSD	Sem.	Subject Area	Type(*)	Borrowed from
Software Architecture Design	6	ING-INF/05	I	A scelta dello studente	D	
Software Testing	6	ING-INF/05	I	A scelta dello studente	D	
Image Processing for Computer Vision	9	ING-INF/03	I	A scelta dello studente	D	LM-TLMD
Gestione Aziendale	9	ING-IND/35	II	A scelta dello studente	D	
Business Processes Automation	3	ING-INF/05	II	A scelta dello studente	D	
Safety Critical Systems	3	ING-INF/05	II	A scelta dello studente	D	
Circuiti per DSP	9	ING-INF/01	I	A scelta dello studente	D	LM-IELM
Instrumentation and Measurements for Smart Industry	9	ING-INF/07	II	A scelta dello studente	D	LM-TLMD
Modelli e Algoritmi di Ottimizzazione	9	MAT/09	II	A scelta dello studente	D	LM-TLMD
Security and Privacy	6	INF/01	I	A scelta dello studente	D	LM-SINF
Ingegneria del suono	6	ING-INF/03	I	A scelta dello studente	D	LM-TLMD
Distributed control and cyber-physical systems design (old name: "*Algoritmi distribuiti e progettazione dei sistemi di controllo su rete*")	6	ING-INF/04	II	A scelta dello studente	D	LM-AUT
Bioinformatica	6	ING-INF/06	II	A scelta dello studente	D	LM-BIO
Quantum Information	6	ING-INF/03	I	A scelta dello studente	D	LM-TLMD
Computer Forensics	6	INF/01	II	A scelta dello studente	D	LM-SINF
Risk Assessment	6	ING-INF/05	II	A scelta dello studente	D	
Cognitive Computing Systems	6	ING-INF/05	II	A scelta dello studente	D	

(*) TEA (Type of Educational Activity)
A = Base (ex 1)
B = Characterising (ex 2)
C = Related or supplementary (ex 4)
D = Activities of your choice (ex 3)
E = Final exam and language skills (ex 5)
F = Further training activities (ex 6 e 7)

TAF	1	2	3	4	5	6	7
rif. DM270/04	Art. 10 comma 1, a)	Art. 10 comma 1, b)	Art. 10 comma 5, a)	Art. 10 comma 5, b)	Art. 10 comma 5, c)	Art. 10 comma 5, d)	Art. 10 comma 5, e)

Calendar of educational activities

	Start	End	Start of exam session for a.a. 2025/2026
1° teaching period	15/09/2025	19/12/2025	**15/12/2025**
Ordinary mandatory exams	1 exam in january, 1 exam in February 2026		
2° teaching period	02/03/2026	12/06/2026	
Ordinary mandatory exams	1 exam in june, 1 in July, 1 in September 2026		
Recovery mandatory exams	1 exam in the period 02/03/2026 - 21/03/2026, 1 exam in the period 19/10/2026 – 14/11/2026		

Course details

Below are the detailed course descriptions for the Master's Degree Program in Computer Engineering.
Students are reminded that instructors are required to update the course descriptions for the subjects they are responsible for on their personal page on the website www.docenti.unina.it , in the section **Teaching** → **Course Descriptions**.

COURSE DETAILS

"ARCHITETTURA DEI SISTEMI DIGITALI"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ALESSANDRA DE BENEDICTIS, NICOLA MAZZOCCA
PHONE:
EMAIL: ALESSANDRA.DEBENEDICTIS@UNINA.IT, NICOLA.MAZZOCCA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II): I
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Knowledge of the design principles of combinational and sequential logic networks and of the general architecture of a computer; programming skills.

LEARNING GOALS

The course aims to provide a methodological and technological approach for the architectural design of dedicated and / or programmable digital systems. The course entails analyzing the design techniques with reference to the development of: microcontrollers, dedicated processors, I / O units, interconnection systems, arithmetic units, units dedicated to IoT and multimedia applications. The activities are carried out with reference to the VHDL language, by means of the use of industrial simulators and FPGA systems (with the related development environments), used for the implementation of the case studies proposed during the course. The main elements for the realization of the documentation and for the testing of digital systems in industrial applications are also presented.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems relating to the design of dedicated and programmable digital systems, with reference to the general project methodologies and the specific constraints deriving from the development tools, programming languages and hardware platforms involved. He/she must also demonstrate to understand the fundamental characteristics of different digital systems models and to be able to identify the most appropriate model or models to solve a specific design problem.

Applying knowledge and understanding

The student must demonstrate to be able to carry out the entire cycle of design and implementation of basic digital systems (e.g. basic combinational or sequential machines) and advanced ones (arithmetic machines, interconnection networks, I / O interfaces, processors), from the architecture and functional behavior specification to the system description by means of the VHDL language, up to the implementation on programmable hardware devices (FPGA), using tools and development environments of wide industrial use.

COURSE CONTENT/SYLLABUS

Digital systems design principles: General purpose, special purpose and embedded systems. Design of digital systems: technological, methodological aspects and supporting environments. The development cycle of a digital system. HDL languages for hardware description and simulation environments.

Recalls and insights on the design of combinatorial machines: Realization of Boolean functions by means of digital circuits. Design parameters and constraints: delays, signal deformations, power dissipations. Minimization of Boolean functions. Design of basic combinational machines (multiplexers / demultiplexers, decoders / encoders, transcoding networks).

Recalls and insights on the design of sequential machines: Models of sequential machines. Impulsive, level-based, synchronous, asynchronous machines. Design of basic sequential machines (registers, counters).

Design of complex systems: Architectural models for the development of complex systems: sequential network systems, pipeline architecture, timing problems, PO / PC model, wired and microprogrammed control systems. Processor design principles. The MIC-1 processor.

Communication protocols and interfaces: Synchronous and asynchronous communication protocols; design of many-to-many interconnection networks; design and use of a serial communication interface according to the RS232 standard.

Architecture and design of arithmetic machines: Parallel and sequential architectures for binary adders, subtractors and multipliers. Application of multiply and accumulate architectures for the implementation of neural networks.
Devices for the synthesis of logic networks: PAL, PLA, FPGA, ASIC. In-depth study of the architecture of the XIlinx FPGAs of the Artix-7 and Spartan 3E families.
Testing of digital systems: Problems and techniques of testing hardware devices.
VHDL language and development environment: Description styles of digital systems and main supporting language constructs. Guidelines for the implementation of basic combinatorial and sequential machines. Process of simulation, synthesis, implementation of a digital design and supporting tools. Timing analysis and supporting tools.

READINGS/BIBLIOGRAPHY

Textbook: Conte, Mazzeo, Mazzocca, Prinetto. Architettura dei calcolatori. Edizioni CittàStudi. 2014. ISBN: 9788825173642.
Lecture notes and presentations provided by the teacher relating to theoretical and applicative topics.
Manuals and datasheets of the devices used for the synthesis of the projects.
VHDL code relating to the exercises carried out in the classroom.

TEACHING METHODS

The course includes about 70% of lectures in which theoretical topics are addressed, while the remaining 30% is reserved for practical lessons and exercises concerning the development of VHDL code for the implementation of specific machines and the use of development environments.
The application part of the course uses professional development tools for which a free license is available and development boards (equipped with FPGAs and various I / O devices) that are distributed to students for the implementation of their own projects.

EXAMINATION/EVALUATION CRITERIA

a) Exam type:

Exam type	
written and oral	x
only written	
only oral	
project discussion	x
other	

In case of a written exam, questions refer to: (*)	Multiple choice answers	
	Open answers	x
	Numerical exercises	

The learning assessment includes a written test consisting of digital systems design exercises and an oral test aimed at verifying the understanding of the theoretical concepts of the course and at discussing a project.

b) Evaluation pattern: none

COURSE DETAILS

"ALGORITMI, STRUTTURE DATI E MACHINE LEARNING"

SSD ING-INF/05*

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS – TEACHER REFERENCES

TEACHER: ROBERTO PIETRANTUONO, CARLO SANSONE
PHONE: 0817683880, 0817683640
EMAIL: ROBERTO.PIETRANTUONO@UNINA.IT, CARLO.SANSONE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): ALGORITMI E STRUTTURE DATI, MACHINE LEARNING
CHANNEL (IF APPLICABLE): N.A.
YEAR OF THE DEGREE PROGRAMME (I, II, III): I
SEMESTER (I, II): ANNUAL
CFU: 12

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to provide first of all the notions necessary for the design and analysis of algorithms and data structures in the development of computer applications. These notions include the theoretical foundations and advanced techniques of design and analysis of algorithms whose application ranges over all aspects related to a processing system, from hardware to software, from operating systems to computer networks, from databases to information systems, from programming languages to software engineering, from human-machine interaction to signal and image recognition, multimedia processing, knowledge engineering, robotics and artificial intelligence and robotics. With reference to the latter field, the course aims to present in detail the main Machine Learning algorithms for the solution of classification, numerical prediction and clustering problems, as well as the methodologies for managing and developing a Machine Learning process, from data preparation to evaluation of results. The course will also allow to develop practical skills in the solution of real problems of classification, numerical predictions or clustering through Machine Learning algorithms, thanks to exercises carried out with open source and / or commercial tools.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

Module of Algorithms and Data Structures

At the end of the course, the student must demonstrate familiarity with a wide variety of known data structures and algorithms that solve fundamental problems, understand the techniques for the synthesis of new algorithms and master the methods to analyze the correctness and asymptotic complexity of algorithms.

Machine Learning Module

At the end of the course, the student must demonstrate knowledge of the main Machine Learning and Deep Learning algorithms and to be able to choose the most suitable Machine Learning algorithm to solve a specific classification and / or numerical prediction and / or clustering problems, based on the requirements of the problem itself. The student must also demonstrate to be able to choose the appropriate data preparation techniques and must know the techniques necessary for the evaluation of the performance of Machine Learning and Deep Learning algorithms.

Ability to apply knowledge and understanding

Module of Algorithms and Data Structures

At the end of the course, the student must demonstrate to be able to apply and combine the main design techniques, as well as advanced data structures, for the synthesis of correct and efficient algorithms for solving problems in the development of computer applications, and to be able to formally analyze the correctness and asymptotic complexity. The training course is oriented to provide the skills and tools necessary to solve new or unfamiliar problems in the broad contexts related to information processing systems.

Machine Learning Module

At the end of the course, the student must demonstrate to be able to solve real problems of classification, numerical prediction or clustering using Machine Learning or Deep Learning algorithms. The student must also demonstrate that he can correctly evaluate the performance of the systems he creates.

PROGRAM-SYLLABUS

Module of Algorithms and Data Structures (6 CFU)

- INTRODUCTORY CONCEPTS (0.25 CFU)Algorithms and data structures, recursion, analysis and design of algorithms.

- ANALYSIS TECHNIQUES AND ELEMENTARY DATA STRUCTURES (1 CFU)
  Correctness analysis: cycle invariant, correctness of recursive algorithms.
  Complexity analysis: asymptotic analysis, O, Ω, Θ notations; analysis of recursive algorithms.
  Basic data structures: dictionaries; piles and tails; priority queues; Lists; hash tables and strings; binary search trees.

- DIVIDE AND RULE, SORTING (1.25 CFU)
  Common problems and algorithms. Sorting: merge sort, heap sort, quick sort.
  Sorting in linear time (counting sort, radix sort, bucket sort), medians and order statistics.

- COMBINATORIAL RESEARCH (0.25 CFU)Exhaustive research, combinatorial research; backtracking, research pruning

- DYNAMIC PROGRAMMING (1 CFU)
  Introduction to dynamic programming. Exhaustive search vs. greedy search vs. dynamic programming. Applications. Common problems and algorithms. String matching, edit distance, longest increasing sequence problem. Fibonacci. Backpack problem. More examples.

- DATA STRUCTURES AND ADVANCED ANALYSIS TECHNIQUES (1 CFU)
  RB shafts, self-adjusting shafts. Graphs. Representation, exploration in breadth and depth, topological ordering. Applications. Advanced algorithm analysis techniques: cushioned analysis.

- COMMON PROBLEMS AND ALGORITHMS, APPLICATION EXAMPLES (1 CFU)
  Number theory problems (e.g. DES and RSA algorithms). Problems on graphs: search for minimum paths. Parallel algorithms, dynamic multithreading. Examples (Fibonacci, sorting). Analysis and comparison with sequential algorithms. Translators and interpreters: lexical analysis, syntactic analysis, data structures used in translators.

- INTRACTABLE PROBLEMS (0.25 CFU)
  Introduction to NP and NP-complete problems. Reducibility. Examples of NP-complete problems.

Exercise part: mainly in C.

Machine Learning Module (6 CFU)

- INTRODUCTION TO MACHINE LEARNING (0.25 CFU)

- INPUT AND OUTPUT (0.5 CFU)Concepts, instances and attributes. Representation of knowledge.

- BASIC METHODS (1.25 CFU)
  Probabilistic models, decision trees, classification rules, linear, instance-based and multi-instance learning models, clustering.

- PERFORMANCE EVALUATION (0.5 CFU)
  Training and Testing. CV, LOO, Cost-sensitive classification. ROC. Evaluation of numerical prediction algorithms.

- ADVANCED METHODS (2 CFU)Decision trees: C4.5. Classification rules. Instance-based learning. Extension of linear models: SVM. Epsilon-SVR. MLP. Numerical prediction with linear models.

- TRANSFORM DATA (0.25 CFU)Attribute selection, PCA, Discretization, Sampling. One-class classification.

- PROBABILISTIC METHODS (0.25 CFU)Bayesian networks. Probability Density Estimation and Clustering. Sequential

and temporal models (outline).

- DEEP LEARNING (0.5 CFU)Training and performance evaluation of deep networks, Convolutional Neural Networks, Autoencoders. Recurrent neural networks and GANs (outline).

- BEYOND SUPERVISED AND UNSUPERVISED LEARNING (0.25 CFU)
  Semi-supervised learning. Multi-instance learning.

- ENSEMBLE LEARNING (0.25 CFU)
  Bagging, Randomization, Boosting, Stacking, ECOC.

Exercise part: Use of open source and/or commercial tools (e.g., Knime, Weka).

TEACHING MATERIALS

Module of Algorithms and Data Structures

TEXTBOOK ADOPTED

1) Thomas Cormen, Charles Leiserson, Ronald Rivest, and Clifford Stein. **Introduction to Algorithms**. 3rd ed. MIT Press, 2009. ISBN: 9780262033848.

TRANSPARENCIES FROM LESSONS AND EXERCISES available on the University professors website on the Microsoft Teams platform.

RECOMMENDED BOOK

2) Steven Skiena. **The Algorithm Design Manual,** 3rd ed, Springer, 2020. ISBN-13: 978-3030542559, ISBN-10: 3030542556.

Machine Learning Module

TEXTBOOK ADOPTED

1) Ian H. Witten, Eibe Frank, Mark A. Hall and Christopher J. Pal. **Data mining: practical machine learning tools and techniques**. 4th ed. The Morgan Kaufmann, 2017. ISBN: 9780128042915.

TEACHING METHODS

Module of Algorithms and Data Structures

The teacher will use: a) lectures for about 80% of the total lesson hours, b) exercises for about 20% of the total lesson hours. The assignment of exercises to be carried out independently and delivered to the teacher is foreseen.

Machine Learning Module

The teacher will use: a) lectures for about 60% of the total hours, b) exercises to deepen theoretical aspects for about 35% of the total hours, c) seminars for about 5% of the total hours.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	X
only written	
oral only	
Discussion of project paper	X
other	X

In the case of a written test, the questions are	**Multiple choice**	
	Free answer	X
	Numerical exercises	

For the module of Data Algorithms and Structures, the exam is divided into n. 2 written tests and an oral test that includes a discussion on the exercises assigned during the course. The written part is divided into n.1 test plus n.1 final test. The tests require, for one or more problems, to implement algorithms for their resolution and analyze their asymptotic complexity.
The oral exam consists of an interview taken after the last written test. It also includes discussion on the resolution of the exercises assigned during the course.

For the Machine Learning module, each student will have to develop 3 projects during the course (so-called mini-contest), which, in case of positive evaluation, exempt him from the development of the final project work.
The oral exam consists of an interview, which also includes the discussion of one of the 3 projects developed during the course or the final project paper.

b) Evaluation methods:

For the module of Algorithms and Data Structures, the 2 written tests each weigh 30% on the final judgment (for a total of 60%). The oral exam weighs for the remaining 40% on the final judgment (with a weight of 30% for the evaluation and discussion of the exercises and 10% for the question on the part of the program not covered by the tests and exercises).

For the Machine Learning module, the discussion of the final project paper (or the project developed during the course) weighs 10% on the final judgment, while the oral exam for 90%.

The final grade of the course will be weighted according to the credits of each module and therefore composed as follows:
• Module of Algorithms and Data Structures, 6 CFU, 50%
• Machine Learning module, 6 CFU, 50%

COURSE DETAILS

"ARCHITETTURA E PROGETTO DEI CALCOLATORI"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: NICOLA MAZZOCCA
PHONE:
EMAIL: NICOLA.MAZZOCCA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I
SEMESTER (I, II): II
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Knowledge of computer architecture, operating systems and communication networks.

LEARNING GOALS

The course aims to provide the methodological, design and technological elements for the design of elaboration systems with reference to pipelined, multi-computer, multi-processor, multi-core and multi-threading architectures. The course also addresses the operation and dimensioning of hierarchical memory systems, the design and programming of I / O units (parallel, serial, DMA and PIC) with the related communication protocols, and the implementation problems of the basic mechanisms for the virtualization of hardware resources (process management mechanisms, virtual machines and hypervisors). Finally, the course presents the main techniques for creating pervasive, autonomic, IoT and edge computing systems, as well as cloud architectures.

The application part of the course is dedicated to the design of I / O drivers and the development of systems operating in the industrial sector. The activities are carried out with reference to applications developed and evaluated experimentally using architectures that involve the use of processing nodes equipped with RISC processors and various I / O devices that can be suitably configured.

With reference to the technological aspects, the architectures of commercial systems for the implementation of industrial applications based on System on Chip or on processing nodes obtained by integrating configurable components are illustrated.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems relating to the design of processing systems, with particular reference to the management of hazards deriving from the use of internal and external parallelism techniques for increasing performance, to the dimensioning of memories, and to the orchestration of different subsystems operating in competition with each other and communicating through different I / O interfaces.

Students must also demonstrate to be able to identify, among the different approaches presented in the course, those that are best suited to specific applications or operating conditions.

Applying knowledge and understanding

The student must demonstrate to be able to design and develop basic software (drivers in the assembly language) necessary to allow communication between different subsystems through the I / O devices presented in the course, even in the presence of concurrent access to common data, as well as concurrency management schedulers. He/she must also be able to carry out the development cycle of medium complexity applications, which require the use of one or more processing nodes, different I / O devices, sensors / actuators, on the supplied hardware devices.

COURSE CONTENT/SYLLABUS

Recalls and insights on processing systems: General purpose and embedded systems. RISC and CISC processors. Wired and microprogrammed control unit. Outage management mechanisms. Introduction to parallelism and pipelining. Review of the Motorola 68000 processor. The MIPS processor: programming model and pipeline. The ARM processor. DSP architectures and applications.

Pipelining and hazards: Techniques for managing data conflicts, hops and interruptions in a pipelined architecture. Superscalar architectures.

Multiprocessor and multicomputer systems: Parallel architectures, speed up and efficiency. Memory coherence algorithms.

I / O peripherals and drivers: Architecture and operation of parallel, serial, DMA and PIC peripherals, and development of drivers for their programming.

The memory hierarchy: Architecture, addressing and sizing of a cache. Virtual memory. Static and dynamic memories.
Bus and interconnection networks: The system bus. Communication protocols. Interconnection networks: multistage switches.
Design and development of systems based on microcontrollers: Principles of design of processing systems for industrial applications based on microcontrollers. Architectures and use of System on a Chip (SoC). **Commercial and industrial programmable devices**. Environments for the design, simulation and analysis of processing systems.
Virtualization and cloud computing. Virtualization techniques and hypervisor. Introduction to cloud systems: service models and applications.
IoT / Edge computing. Architectures and applications of IoT and edge computing systems. Development of commercial edge systems and integration of sensor networks.

READINGS/BIBLIOGRAPHY

Textbook: Conte, Mazzeo, Mazzocca, Prinetto. Architettura dei calcolatori. Edizioni CittàStudi. 2014. ISBN: 9788825173642.
Lecture notes and presentations provided by teachers relating to theoretical and applicative topics covered in the course.
Manuals and datasheets of the devices used for the implementation of applications.

TEACHING METHODS

The course includes about 70% of lectures in which theoretical topics are addressed, while the remaining 30% is reserved for practical lessons and exercises concerning the development of I / O drivers and the use of development environments.
The application part of the course uses professional development tools for which a free license is available and development boards (equipped with an ARM microcontroller and various I / O devices) that are distributed to students for implementation of their projects.

EXAMINATION/EVALUATION CRITERIA

c) Exam type:

Exam type	
written and oral	x
only written	
only oral	
project discussion	x
other	

In case of a written exam, questions refer to: (*)	**Multiple choice answers**	
	Open answers	x
	Numerical exercises	

The learning assessment includes a written test consisting of design exercises for systems based on I / O devices and an oral test aimed at verifying the understanding of the theoretical concepts of the course and discussing the exercises implemented on the development board.

COURSE DETAILS

"NETWORKS & CLOUD INFRASTRUCTURES"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: GIORGIO VENTRE
PHONE:
EMAIL: GIORGIO.VENTRE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the Regulations of the CdS)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide the necessary advanced methodological and operational skills on the design and management of complex computer networks and on computing architectures distributed on the Internet such as Cloud Systems and CDN architectures. The training objectives are to provide: the concepts of transmission with guarantee of quality of service; the internal architectures of network systems; the main interconnection architectures both for metro and wide-area networks, and within data-centers and HPC systems; technologies and methodologies for traffic engineering; problems related to internetworking through complex and multidomain structures; advanced techniques for both intra-domain and inter-domain routing; the creation of network services and applications based on the Software Defined Networks approach; network design strategies with reliability characteristics; the definition and control of Service Level Agreement; the basic concepts for Cloud Computing; Cloud service models (IaaS, PaaS, SaaS). The activation models of Cloud services (public, private, hybrid, community); the scalability of Cloud services; Network Function Virtualization.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge of the concepts related to the quality of service in terms of control and management mechanisms. Must be able to understand the difference of the various architectural solutions for the construction of network infrastructures in the various scenarios covered in the course and must have skills in the mechanisms for defining metrics and routing policies. Must have knowledge of Traffic Engineering techniques also with regard to the definition of SLA. It must be able to understand the different solutions to activate and adopt services based on Cloud architectures.

Ability to apply knowledge and understanding

The student must demonstrate to be able to apply the techniques learned for the solution of advanced problems of configuration of networks and cloud services and definition of traffic engineering strategies.

PROGRAM-SYLLABUS

Part I –Network System Architectures

- Switched networks. Switching systems. Input-queued crossbar switches and Head-of-Line blocking. Virtual output queues. Multi-stage interconnection networks. Clos theorem. Spanning Tree Protocol (STP). TRILL.
- Virtual networking. Ethernet VLANs. Virtual network interfaces: TUN/TAP, MacVLAN, and MacVTap. Virtual Ethernet Port Aggregator (VEPA). SR-IOV NICs.
- Software switches: Linux Bridge, Open vSwitch.
- Router architectures and operating systems. Software routers: XORP, Quagga, FRRouting.
- Bare metal, whitebox network switches.
- Data Plane and Control Plane issues. SDN. SDN southbound protocols: OpenFlow. SDN controllers.
- Quality of Service concepts. QoS control and management in packet switched networks.

Part II – Network Infrastructures

- Intra-domain routing. OSPF. IS-IS. ECMP. ECMP in OSPF.
- Inter-domain routing. BGP. SD-WAN. SD-WAN solutions.
- IP networks and virtual circuits. MPLS & switched networks. Traffic Engineering in MPLS networks. RSVP-TE. Segment Routing. Path Computation Element (PCE).
- Evolution of access and carrier networks. Optical networking. SONET/SDH. DWDM. PON and GPON.
- Carrier Ethernet. Metro Ethernet. Ethernet service over MPLS. Ethernet Virtual Private LAN Service (EVPLS).
- Content Delivery Networks.
- Resiliency of Networked Infrastructures. Principles of Fault Tolerant Design.

Part III – Cloud Infrastructures

- Cloud Computing: foundational concepts. Cloud service models (IaaS, PaaS, SaaS). Cloud deployment models (public, private, hybrid, community).
- Elastic computing. Horizontal vs vertical scalability in the cloud. Cloud resources and identity.
- Service Level Agreements. Billing models for cloud services.
- Network Function Virtualization (NFV). The NFV placement problem and solution methods.

TEACHING MATERIALS

- Larry Peterson & Bruce Davie, Computer Networks - A system approach. Sixth Edition, Morgan Kauffman, 2021 – ISBN: 978-0128182000
- Lesson slides

TEACHING METHODS

The course consists of: a) lectures for about 80% of the total hours; b) practical exercises for the remaining 20%.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

The exam consists of an oral test and the discussion of a projectpaper.

COURSE DETAILS

"IMPIANTI DI ELABORAZIONE"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: DOMENICO COTRONEO
PHONE: 0817683824
EMAIL: DOMENICO.COTRONEO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Knowledge of computer architecture, operating systems and communication networks.

LEARNING GOALS

The aim of the course of Processing Plants is to provide the methodological elements for the analysis and quantitative evaluation of modern processing systems. Particular attention will be given to both performance characteristics and those of safety and reliability.

In order to achieve its objectives effectively, the course presents, from an exquisitely engineering perspective, the following topics:

- methodologies for analyzing the performance and security of systems, such as inferential statistics, *experimental design*, simulation and stateless and stateless analytical models;
- methodologies and techniques for direct measurement of large-scale systems, such as data centers, with particular emphasis on *field data analysis* methodologies and techniques;
- methodologies and standards for the design and implementation of *safety critical* systems.

The application part of the course is dedicated to the evaluation of the performance and reliability of some systems, performed during the lessons; and data analysis aimed at evaluating the reliability and security of real large-scale systems (data centers or High Peformance Computing systems).

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems for the analysis of the performance and reliability of the systems, with particular reference to statistical evaluation and *experimental design* techniques, and to direct measurement techniques. The student must also demonstrate knowledge of the fundamentals of *safety-critical* systems engineering: *hazard* analysis; identification and risk management; techniques for identifying failures.

Ability to apply knowledge and understanding

The student must demonstrate to be able to evaluate the performance and reliability of a processing system, to compare the performance and reliability of two or more systems, to know the main statistical techniques for data analysis, coming from large-scale systems (data centers or systems for High Peformance Computing)

PROGRAM-SYLLABUS

CONTENTS

Fundamentals of Inferential Statistics applied to the engineering of processing systems. Sample distributions. Confidence intervals. Hypothesis testing. Regression models. Linear regression. Design Of Experiments. Simple Design. Full Fractorial Design. Fractional Factorial Design. ANOVA. Brief notes on statistical data processing tools: Matlab (statistical toolbox) and JMP.

Methodologies and Techniques for Experimental Analysis. . Capacity Planning and Capacity Management. Workload types. Application Benchmarks. Criteria for selecting workloads. Workload characterization techniques: Averaging, Single Parameter Histograms, Multi-parameter Histograms Principal Component Analysis, Markov Models and Hierarchical Clustering. Capacity Test .

Analysis of the performance of a processing plant. Models for performance measurement and models for measuring reliability attributes. Analytical Models and Simulation Models. Discrete-time Markovian models. Markovian models in continuous time. Queue theory.. Mean Value Analysis algorithm. Exercise on some case studies.

Analysis of the reliability of a processing plant. Repairable and non-repairable systems. Definition of Dependability: Availability, Reliability, Safety. Dependability measures. Fault, Error and Failure. Fault avoidance techniques and Fault Tolerance techniques. Duplication. N-Modular Redundancy. Hardware redundancy. Notes on Fault Tolerance software strategies (N-Version and Recovery Block). Notes on the security of cyber physical systems.

Models for assessing reliability. Combinatorial models: Reliability Block Diagrams and Fault Trees. Modelling and evaluation of a TMR. Examples of modeling of real systems (e.g. banking communication networks and aircraft).

Field Failure Data Analysis Methods and Techniques: FFDA in the context of data mining. Data Collection. Data Filtering. *Data correlation and Coalescence*. Failure analysis of large-scale systems. Real case studies. Support for data analysis with Python language
Safety critical systems. Definition of Hazard and Risk. Hazard Analysis. The FMEA. Risk analysis and management. Notes on Safety Standards: IEC61508, DO178B, CENELC EN 50128.
Notes on the management of a processing plant: The writing of a technical specification. The testing of a plant. Aspects characterizing the management of a processing plant. *Business continuity plan* and *Disaster Restoration* Techniques. Backup *and Restore* Techniques

TEACHING MATERIALS

Textbooks:

- Raj Jain, The art of Computer systems Performance Analysis, Wiley
- Douglas C. Montgomery, George C. Runger - Applied Statistics and Probability for Engineers-Wiley
- KishorTrivedi, Andrea Bobbio Reliability and Availability Engineering, Cambridge University Press

Teaching handouts (available from the course website)
Handouts and presentations provided by the teacher related to theoretical and applicative topics covered in the course.

TEACHING METHODS

The course includes about 70% of lectures in which theoretical topics are addressed, while the remaining 30% is reserved for practical lessons and exercises concerning the application of the acquired techniques to real systems. The application part of the course uses tools for statistical data processing, such as Matlab, R and JMP.

EXAMINATION/EVALUATION CRITERIA

d) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	x
Discussion of project paper	x
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

COURSE DETAILS

"TRASMISSIONE DEI SEGNALI DIGITALI"

SSD ING-INF/03

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: DAVIDE MATTERA
PHONE: 0817683795
EMAIL: DAVIDE.MATTERA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I
SEMESTER (I, II): I
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to introduce the theme of the transmission of information between two points. The course aims to provide students with specialized notions on the digital transmission of information, which plays a critical role in any information processing system, and on the evaluation of engineering parameters that briefly describe the quality of information transfer.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems related to the synthetic description of the waveform channel, frequency conversion, and signaling and reception on a waveform channel. It must demonstrate that it can elaborate arguments concerning the links between the processing of the received signal and the probability of error that characterizes the information transfer, remember its link with the strength of the background noise that overlaps the received signal, illustrate the link between the bit-rate at which the information transfer takes place and the band available on the waveform channel.

Ability to apply knowledge and understanding

The student must demonstrate to be able to solve problems concerning the evaluation of the probability of error starting from the definition of a specific mechanism of information reception, also using tools dedicated to computer processing of information signals, and to build processing software by manipulating standard processing schemes studied during the course.

PROGRAM-SYLLABUS

- Description of waveform channel and equivalent low-pass signals (1 CFU)
- Description of the fundamental mechanisms to achieve frequency conversion (1 CFU)
- Non-deterministic description of signals (2 CFU)
- Digital signaling on AWGN channel (3 CFU)
- Digital signaling on limited bandwidth channel (2 CFU)
-

TEACHING MATERIALS

Course handouts written by Prof. Mattera and made available online to the students of the course
Proakis-Salehi, Digital Communications, McGraw-Hill Education, 2007

TEACHING METHODS

The course consists of lessons for a total of 72 hours of which 48 hours are dedicated to the definition of theoretical topics and 24 hours are dedicated to exercises that make use of specialized software and aim to deepen the understanding of theoretical topics.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	
other	

COURSE DETAILS

"TEORIA DELL'INFORMAZIONE"

SSD ING-INF/03

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: AUBRY AUGUSTO
PHONE:
EMAIL: AUGUSTO.AUBRY@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): *TEAMS: "TEORIA DELL'INFORMAZIONE-ANNO ACCADEMICO 2021/2022"*
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): I
CFU: 9

PREPARATORY COURSES (if required by the Regulations of the CdS)

The course is tailored to students who have obtained a first-level degree in Information Engineering or Computer Science.

PREREQUISITES (IF APPLICABLE)

None.

LEARNING GOALS

The main objective is the quantitative definition of information, so as to enable the student to understand the criteria underlying the design of information storing, processing and transmission systems. The student will become familiar with such concepts as "redundancy removal" (a.k.a. source coding or lossless compression), lossy compression (a.k.a. quantization, whether scalar or vector), as well as with such concepts as channel coding and channel capacity. A relevant objective is also to illustrate some codes - mainly source codes - of relevant practical importance, such as arithmetic coding and universal coding. On a parallel track, the course develops basic concepts of inferential statistics and optimization.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student will acquire a clear understanding of the reasons for uncountable design choices made in the context of information storing, processing and transmission systems, and of the basic concepts underlying the design of source and channel codes. Also, they will become familiar with some basic techniques underlying data analysis and system optimization.

Ability to apply knowledge and understanding

The student will be able to solve simple resource allocation problems and to analyze some information processing/compression/transmission systems.

PROGRAM-SYLLABUS

General concepts. Information and its measures. Entropy, joint and conditional entropy. Mutual information. Divergence.
Information sources. Models for information sources. Chain rule for entropy and entropy rate.
Markov chains. Law of large numbers and typicality. Compression.
Lossless Compression. Source codes. Kraft inequality and its consequences. Fundamental limits for lossless compression. Shannon, Huffman and Shannon-Fano-Elias codes. Arithmetic coding. Lempel-Ziv coding.
Lossy compression. Distortion. Hamming and quadratic distortion. Rate-distortion function and its interpretations. Rate-distortion function for relevant cases. Extending the idea of typicality and rate-distortion theorem.
Channel capacity. Definition. Capacity of some elementary channels. The channel capacit theorem: direct and converse propositions. Some basic concepts of Channel coding.
The Gaussian Channel. Additive Gaussian channel. The Shannon theorem for the Gaussian channel. Parallel Gaussian channels and waterfilling. Colored Gaussian channels.
Elements of Inferential Statistics. What is inference. Prior and posterior characterization.
Elements of Bayes and non-bayes statistics. Cramer-Rao limits. LMMSE estimation. Gradient Algorithm.
Elements of Optimization Techniques. What is a constrained optimization problem. Basics on optimality conditions. Total least squares.

TEACHING MATERIALS

T. M. Cover, J. A. Thomas, *Elements of Information Theory*, 2nd Edition, J. Wiley &Sons, 2006.
Slides and notes will be made available on the dedicated MSTeams channel.

TEACHING METHODS

The course consists of approximately 70 class hours. 2/3 of the hours will be devoted to theory, 1/3 to numerical exercises.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test

written and oral	
only written	
oral only	X
Discussion of project paper	
other	

b) Evaluation methods:

The evaluation will take place on the basis of an oral interview aimed at verifying that the student has matured the basic concepts taught during the course.

COURSE DETAILS

"ARCHITETTURA DEI SISTEMI INTEGRATI"

SSD ING-INF/01

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ANTONIO G. M. STROLLO
PHONE: 0817683125
EMAIL: ANTONIOGIUSEPPEMARIA.STROLLO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I
SEMESTER (I, II): I
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Basic knowledge of digital circuits, the main characteristics of MOS devices and CMOS logics.

LEARNING GOALS

As part of the course, the design flow of digital integrated circuits is studied, starting from the description using hardware description languages to the physical implementation. The course aims to provide students with the methodologies and knowledge useful for designing modern high-scale integration microcircuits, evaluating their characteristics, optimizing their performance and defining their verification procedures.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course aims to provide students with the knowledge and methodological tools necessary to complete the development flow of an integrated digital system. In particular, the student will be informed about the standards used to create descriptions that can be synthesized using languages for the description of the hardware, on the techniques for evaluating and optimizing delays and power dissipation, on the various architectures for the implementation of arithmetic circuits and on the main testing methodologies. These tools will allow students to understand the main relationships between the physical implementation of integrated systems and their electrical characteristics.

Ability to apply knowledge and understanding

At the end of the learning process the student is able to design and analyze at an architectural, circuit and physical level high-scale digital circuits and systems of integration. He will have the methodological and operational tools necessary to describe a digital system through HDL languages, to prepare the test-bench and to carry out the simulation. The student will also be able to define the appropriate constraints necessary to carry out the synthesis phase, to use automatic synthesis programs and to evaluate the results provided by the synthesizer. The training course is aimed at transmitting the methodological and operational skills and tools necessary to concretely apply the various techniques aimed at optimizing the performance of digital systems, in terms of speed, occupied area and energy consumption.

Making judgements, Communication skills, Learning skills

At the end of the course the student must be able to independently choose the possible methodologies and techniques to be used in the various phases of the design cycle of a high-scale digital integration system; They must also have the ability to evaluate the results deriving from the application of various optimization techniques and be able to compare the performance of different architectures.

Students will be able to independently deepen topics covered. The verification methodology and the comparison with the teacher also tend to develop the communication skills of the students who must demonstrate that they know how to set up a scientific relationship using appropriate terminology and language.

PROGRAM-SYLLABUS

Classification of integrated systems: full-custom, based on standard and programmable cells. Design methodologies of integrated systems. Automatic synthesis techniques and nesting and connection of standard cells.

Features of the latest generation MOS transistors. Switch-level simulation techniques. Simplified assessment of logic gate delays.

Static analysis of delays. Delay graphs. Characterization of standard cell delays. Interconnection levels and parasitic parameters. Evaluation of delays introduced by interconnections using Elmore's technique. Repeaters. Effects of technology scaling on interconnection delays. Cross-talk. Distribution of supply lines.

Design and timing of sequential systems. Characteristic times of the registers. Advanced registers. Pipelining techniques. Effects of non-idealities of the clock (skew, jitter) on the timing of sequential systems. Clock generation and distribution. Phase coupling (PLL) and delay hook (DLL) rings.

Evaluation of power dissipation in VLSI systems. Sources of static and dynamic power dissipation. Techniques for reducing power dissipation at a technological, circuit and architectural level.

The VHDL language for the description and synthesis of integrated systems. Sequential and concurrent statements. Event-

driven simulation. Standard libraries for the synthesis of digital systems. Description and synthesis of arithmetic circuits. Test-bench. Text file operations.
Testing of integrated CMOS systems. Fault patterns. Algorithms for calculating test vectors. Self-testing techniques. Arithmetic circuits. Delay selection adders, carry-skip, parallel-prefix. Multi-operating adders. Parallel multipliers. Fast multipliers (Wallace, Dadda).

TEACHING MATERIALS

- Weste, Harris, "CMOS VLSI Design – circuit and systems perspective", 4th edition, Pearson – Addison Wesley, 2011
- Lecture notes
- Tutorial texts

TEACHING METHODS

The course includes lectures, exercises and, compatibly with the organizational aspects, laboratory exercises.
Students adopt Cadence's integrated systems development programs to carry out the exercises, made available through the "Cadence Low-Cost Classroom Teaching Licenses" acquired through Europractice. Students also use an open-source vhdl (ghdl) simulator and waveform viewer (gtkwave).

EXAMINATION/EVALUATION CRITERIA

a) Examination method:

The exam is divided into a test	
written and oral	
only written	
oral only	✓
Discussion of project paper	
Other (Tutorial discussion)	✓

COURSE DETAILS

"ALGORITMI DI OTTIMIZZAZIONE COMBINATORIA SU RETE"

SSD MAT/09

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: MAURIZIO BOCCIA
PHONE: 0817683247
EMAIL: MAURIZIO.BOCCIA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I
SEMESTER (I, II): II
CFU: 9

PREPARATORY COURSES

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to provide students with the methodological tools to analyze and solve mathematical programming problems with particular reference to combinatorial optimization and network optimization problems. Solving methodologies are presented both for some "basic" problems of optimization on networks, and for "difficult" problems of combinatorial and network optimization. At the end of the course the student will have acquired the ability to formulate an abstract model of a combinatorial optimization problem, the ability to identify the structures present on which to articulate a heuristic solution approach, or to apply an ad hoc algorithm for its exact solution.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course aims to provide students with the methodologies of combinatorial optimization and network optimization necessary for the modeling and exact resolution and / or heuristic of decision problems that may arise in engineering and industry. At the end of the course, the student will have to demonstrate knowledge and ability to use the tools necessary to formulate a combinatorial optimization and optimization problem on networks. It must also be able to decide, in relation to the characteristics and complexity of the problem, whether the problem can be solved through the use of the most popular software packages and optimization libraries, or it is necessary to develop an ad hoc algorithm for its solution. He must therefore demonstrate knowledge of the main meta-heuristic methodologies for solving combinatorial optimization problems and be able to choose the most suitable methodology for the problem to be addressed.

Ability to apply knowledge and understanding

The training course is aimed at transmitting to students the methodological and operational tools necessary for the formulation and solution of combinatorial optimization problems and optimization problems on networks. In particular, the student must demonstrate the ability to formulate an optimization problem. It must also demonstrate that it can solve it, whenever possible, exactly using one of the main optimization libraries, or in an approximate manner using a heuristic approach. Finally, he/she must be able to implement and test the different heuristic and meta-heuristic approaches in order to decide on the most suitable approach to the particular problem.

PROGRAM-SYLLABUS

Models of Operations Research

- The modelling approach
- Optimization Models

Continuous and integer linear programming

- Introduction to Linear Programming
- Examples of linear programming models
- Graphical representation of a P.L. problem
- Formulations.
- The Branch and Bound method
- The cutting plane method
- The Branch and Cut method

Software for Mathematical Programming

- Use of the Gurobi library for the solution of PL and PLI problems.
- Examples of library use: production planning, inventory management, network problems, localization problems, distribution problems.

Heuristics for the solution of combinatorial optimization problems

- Combinatorial optimization
- Greedy heuristics
- Local search and taboo search
- Genetic algorithms

Elements of graph theory

- Forms of representation of a graph
- Algorithms for visiting a graph

The problem of routing and flows on the network

- Classification of problems and minimum path algorithms
- The model of the minimum path
- Algorithms for calculating minimum routes
- Single and Multi-Commodity Flow Issues
- Network routing and balancing policies

Design, localization, and clustering issues

- Localization problems on nodes and arcs
- Constructive and improving heuristics
- The clustering problem
- k-means algorithm for clustering problem

The Salesman's Problem (TSP)

- Complexity of the problem
- A row generation algorithm
- Greedy heuristics and local search
- The problem of orienteering as a variant of the TSP

The Problem of Vehicle Routing

- Definition of the problem and its variants
- Greedy heuristics and local search

TEACHING MATERIALS

- M. Caramia, S. Giordani, F. Guerriero, R. Musmanno, D. Pacciarelli, "Ricerca Operativa", Isedi, Italy, 2014.
- H. Paul Williams, Model Building in Mathematical Programming, John Wiley & Sons, Ltd.

- F. S. Hillier, G. J. Lieberman, Operations Research - Fundamentals, 9/ed., McGraw-Hill, 2010.
- A. Sforza, Models and Methods of Operations Research, 3rd ed., ESI, Naples, 2018.
- G. Bruno, Operations Management. Models and methods for logistics, ESI – Edizioni Scientifiche Italiane.
- Supplementary teaching material provided during the course.

TEACHING METHODS

The teacher will use: lectures (65%), seminars (5%), numerical exercises (10%), exercises using optimization libraries (25%). The course material will be made available online to students.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	X
only written	
oral only	
Discussion of project paper	X
other	

In the case of a written test, the questions are	**Multiple choice**	
	Free answer	X
	Numerical exercises	X

(b) Evaluation procedures:

The written test is aimed at verifying the ability to formulate optimization problems and the understanding of PL and PL problem-solving algorithms. The student has 2 hours for the written test. The exam also includes the development of a project in which the student must implement and experiment an exact and / or heuristic algorithm for the solution of a combinatorial optimization problem. The oral interview will have as its object both the discussion of the project and the assessment of the acquisition of the concepts and methodologies illustrated during the lessons.

The outcome of the written test is binding for access to the oral exam. The written and oral tests with the description of the final paper contribute respectively for 40% and 60% of the final evaluation. Passing the written test is not sufficient to pass the exam.

COURSE DETAILS

"CALCOLO SCIENTIFICO PER L'INNOVAZIONE TECNOLOGICA"

SSD MAT/08

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: LUISA D'AMORE
PHONE: 081675626
EMAIL: LUISA.DAMORE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I
SEMESTER (I, II): II
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

Computer numerical simulations (which Galileo Galilei called mathematical and experimental analysis of nature) allow us to master the increasing complexity of science and engineering models: by exploring innovative solutions through the intensive use of emerging hardware and software technologies, they have a decisive social impact (policy decision-making). Numerical simulation is the result of an ever-changing process. Reliable simulations, i.e. simulations that are able not only to interpret but also to predict, must take uncertainty into account. The analysis of uncertainty, its propagation and the study of the sensitivity of the solution to uncertainties in the data (Uncertainty Quantification) is a fundamental requirement of predictive simulations.
The course aims to deepen methodological knowledge of numerical mathematics (at the base of numerical simulations) preliminary and fundamental to calculate, interpret and describe the solutions. The critical analysis of numerical methods, solution algorithms and their implementation in computing environments such as Matlab (also Parallel), will allow students to identify the best approach to obtain the expected result.

The LEARNING GOALS of the course are the following

- Acquire specialized methodological knowledge of the *mathematics stack* for computer simulations and modeling (from problem formulation to interpretation, visualization, verification and validation of results;
- Deepen the tools of numerical mathematics (algorithms and software), selecting them appropriately according to the type of problem and the calculation tool (co-design);
- Master the numerical skills necessary to deal with the modeling and computational simulation of complex systems of interest in Computer Engineering in an interdisciplinary and multidisciplinary context.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course aims to provide students with the methodological knowledge of scientific computing necessary for the realization of scientific simulations of problems that arise in very different production realities. The student must demonstrate knowledge of the aspects characterizing the quantification of uncertainty in computational simulations or understand the problems related to the main sources of error that arise in the computational resolution phase, to recognize and reduce their propagation on the numerical solution through the critical analysis of numerical algorithms and to develop reliable software elements able to estimate the uncertainty on the result. At the end of the course, the student will be able to define the process of computational solving of some fundamental problems of numerical mathematics.

Ability to apply knowledge and understanding

The training course is oriented to provide the operational skills necessary to transform any mathematical model, such as a partial differential equation, an integral or a system of linear equations, into an element of mathematical software and to use quality standards for the verification and validation of results.
At the end of the course, the student will be able to select and use the tools of numerical mathematics (methods, algorithms and software) acquired to solve some case studies by analyzing, interpreting and validating the results obtained.

PROGRAM-SYLLABUS

Structure of the course:
Part 1 - 30 % – basic concepts
Part 2 – 20 % – insights
Part 3 – 50 % – preparation for the final exam

Part 1:

Introduction to the principles of Scientific Computing – computational solving phases of a problem, sources of error,

stability of an algorithm, conditioning of a problem. Error propagation analysis.

Numerical Linear Algebra – Direct methods, Gauss Elimination Algorithm and LU factorization for full or structured matrices.

Part 2:
Specialized Linear Algebra - Iterative methods. The Preconditioned Conjugate Gradient method. The MULTIGRID methods. The Basic Linear Algebra Subroutines (BLAS) and Parallel BLAS (PBLAS) libraries. Introduction to the problems related to the development of parallel algorithms for matrix calculation and performance analysis (speed up, efficiency). Experimentation in Matlab's Parallel Toolbox environment.

Part 3:
Data representation – Interpolation and approximation methods for data processing and analysis. Spline functions. Interpolation of curves using parametric splines. Bezier curves, B-spline- NURBS curves. Representation of objects in two and three dimensions. The polynomial of least squares of first and second degree. Introduction to numerical models for machine learning. Introduction to software environments for object modeling.

FFT algorithm – From the Fourier transform to the discrete Fourier transform using numerical quadrature formulas. FFT algorithm of Gentleman and Sande, FFT algorithm of Cooley and Tukey. Examples and applications in Matlab: filtering/denoising/reconstruction of a signal and an image. Radix-2, radix-r, mixed-root FFT algorithm.

Partial Differential Equations: Preliminary concepts and definitions on partial differential operators: **c**laxification (of operators, initial and boundary conditions, partial differential problems); Examples in application contexts. The numerical approach underlying the resolution of partial differential problems; Introduction to discretization methods: finite differences and finite volumes; Concept of consistency, stability and convergence.

TEACHING MATERIALS

- A.D'Alessio, Lezioni di Calcolo Numerico e Matlab- Liguori editore, IIed
- L. D'Amore, Gleanings on the method of Conjugated Gradients, University Library.
- A.Murli, Numerical Mathematics, part one. Ed. Liguori

Handouts on specific topics available on the teacher's website

TEACHING METHODS

The teacher will use
a) Lectures for about 30%
b) Exercises for about 20%
c) Laboratory for about 20%
d) Specialized software for about 30%

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written or halfway through	x
oral only	
Discussion of project paper	x
Other	

In the case of a written test, the questions are (*)	Multiple choice	
	Free answer	
	Numerical exercises	x

VERIFICA DI APPRENDIMENTO E CRITERI DI VALUTAZIONE

b) Modalità di esame:

L'esame si articola in prova	
scritta e orale	
solo scritta o intercorso a metà	x
solo orale	
discussione di elaborato progettuale	x
Altro	

In caso di prova scritta i quesiti sono (*)	A risposta multipla	
	A risposta libera	
	Esercizi numerici	x

COURSE DETAILS

"ELABORAZIONE DI SEGNALI MULTIMEDIALI"

SSD ING-INF/03

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: LUISA VERDOLIVA
PHONE: 0817683929
EMAIL: LUISA.VERDOLIVA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide knowledge of the basic concepts and algorithms for digital image processing and present the main techniques for encoding still images and video sequences, with particular attention to the most common standards. In addition to providing the mathematical and conceptual tools to analytically deal with these topics, the course aims to give the necessary knowledge to develop the main algorithms for image processing in Python.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the methodological tools for image analysis and processing. Such tools will allow students to solve more complex problems in both the space domain and the frequency domain.

Ability to apply knowledge and understanding

The student must demonstrate that he knows how to reason about the problems regarding the analysis and processing of images and to be able to choose the most suitable technique for solving a practical problem.

PROGRAM-SYLLABUS

Image enhancement. Enhancement in the spatial domain. Basic transformations of intensity: linear and nonlinear point operations. Histogram equalization. Bit-plane slicing. Arithmetic. Geometric operations. Basics of spatial filtering. Smoothing and sharpening filters. Median filter. Morphological operations. Enhancement in the frequency domain. Two-dimensional Fourier transform. DFT-2D. Examples of low-pass and high-pass filters. Frequency filtering: ideal filters, Butterworth filters and Gaussian filters.

Color representation. Notes on the human visual system. Cones and rods. Relative sensitivity of S, M and L cones. Trichromatic color theory. Color matching. Color spaces (RGB, HSI). Pigments: subtractive coloring, CMY printing systems, CMYK (CMYK printing).

Segmentation. Edge based techniques. Point detection and line detection. Gradients of Roberts, Prewitt and Sobel. Gradient thresholding. Zero-crossing of the Laplacian. Canny edge detector. Class-based techniques. K-means algorithm.

Image compression. Source encoding. General information about data compression. Quantization. Uniform and non-uniform quantization. Predictive coding. Diagram of the encoder and decoder. Predictive quantization. Transformed encoding. Energy compaction and optimal allocation of resources. Linear transforms. Karhunen-Loéve transform and its properties. Discrete cosine transformed. The JPEG standard.

Video encoding. General information about the video signal. Spatial and temporal compression. Conditional replenishment and motion compensation. The hybrid encoder. Introduction to MPEG-1 and MPEG-2. Scalability in resolution and frame-rate

Wavelet transform. Time-frequency localization. Continuous wavelet (CWT). Mother Wavelet. Multi-resolution analysis, scaling function. MRA equations. Extension to the two-dimensional case. Implementation of the discrete wavelet transform (DWT). Bank of analysis and synthesis filters. Encoding using Wavelet. EZW algorithm.

Applications. Examples of advanced applications for image processing: denoising, super-resolution, recognition of faces or objects, classification using local descriptors, semantic segmentation, recognition of manipulations in images and videos also with learning-based techniques (outline of convolutional neural networks).

TEACHING MATERIALS

Recommended textbooks:

- R.C.Gonzalez, R.E.Woods: "Digital image processing", 3nd edition, Prentice Hall, 2008.
- A.Bovik: "The essential guide to image processing", Academic Press, 2009.
- K.Sayood: "Introduction to data compression", 2nd edition, Morgan Kaufmann, 2000.

Multimedia Signal Processing course notes: http://wpage.unina.it/verdoliv/esm/

TEACHING METHODS

Teaching is provided: a) for about 70% with lectures; b) for about 30% with guided laboratory exercises for the development of software applications in Python to better understand the techniques studied.
The topics of the lectures and exercises are exposed with the help of electronic whiteboards and / or detailed transparencies, made available to the student in the teaching material through the official website of the teacher. Lessons are also recorded.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	X
only written	
oral only	
Discussion of project paper	
Other	

The exam is divided into a written and an oral test. The written test, which consists of three algorithms to be developed in Python on the computer, can be replaced by the development of a practical project in Python on an advanced image processing application. The oral exam consists of two questions on problems/algorithms exposed to the course.

COURSE DETAILS

"INFORMATION SYSTEMS AND BUSINESS INTELLIGENCE"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: FLORA AMATO
PHONE: 0817683835
EMAIL: FLORA.AMATO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to provide specialized skills for the design and management of modern Business Intelligence (BI) infrastructures to support directional Information Systems and seen both as a tool at the service of achieving business objectives, and as a catalyst for organizational and strategic innovation. In particular, some of the most popular open-source and commercial suites for BI will be analyzed and Data Mining methodologies and techniques for the extraction of information and knowledge useful for decision-making processes will be deepened. In addition, modern ERP/CRM technologies supporting operational business processes will also be detailed, and how these are one of the main data sources in a BI system will be analyzed. Finally, the methodological principles of some phases of the life cycle of an Information System are addressed, with reference not only to the technological aspects, but also to those that require attention to the organizational and economic context, as well as the related assessment and benchmarking issues.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course aims to provide students with specialized knowledge and all the methodological and technological tools to support both Management Information Systems and Operational Information Systems, with particular reference to the main ERP / CRM and Business Intelligence (BI) software suites, both commercial and open-source. In particular, these tools will allow students, on the one hand, to know how to use what are the most appropriate "best practices" for the standardization and re-engineering of business processes of an operational nature, on the other, to know how to efficiently and effectively implement the directional / decision-making processes of a given organization.

Ability to apply knowledge and understanding

The training course is aimed at transmitting the methodological and operational skills and tools necessary to concretely apply the knowledge related to the design and maintenance of both Executive and Operational Corporate Information Systems. In particular, the student must demonstrate to be able to re-engineer through the introduction of ERP / CRM information technologies those that are the typical operational business processes of a company, and to support with Business Intelligence (BI) tools those that are instead the directional / decision-making processes.

PROGRAM-SYLLABUS

1. **Business Information Systems**: Introduction to Business Information Systems. The organizational, functional and IT model of an Information System. Operational Information Systems vs Management Information Systems.
2. **Management Processes in Information Systems**: Types of process classification. Identification, description and breakdown of processes. Notes on Process Modeling and Workflow. Performance of Management Processes. Business Process Reengineering (BPR).
3. **Information Technologies at the base of Operational Information Systems**: Architecture of modern Information Systems. ERP systems. CRM systems. SOA architectures. Integration of Information Systems.
4. **Management Information Systems and Business Intelligence**: Definition of Business Intelligence (BI). The BI process. Main commercial and open-source tools. Data Mining techniques to support BI.
5. **Life Cycle of Information Systems**: Planning. Assessment and Benchmarking. Re-engineering and Feasibility Study. Design, Construction and Maintenance. Management and Management. Notes on Project Management.
6. **Examples of Information Systems**: Information Systems for Logistics and Production. Transport Information Systems. Health Information Systems. Information Systems for Public Administration. Geographic Information Systems.

TEACHING MATERIALS

Recommended textbooks:

- Bracchi, Francalanci, Motta, "Information systems for the digital industry", Mc Graw Hill 2010.
- Rezzani, "Business intelligence. Processi, metodi, uso in azienda", Apogeo, 2012.

Teacher's notes, Lucidi.
Manuals of the various tools presented.

TEACHING METHODS

The teacher will use lectures for about 60% of the total hours, and in addition computer exercises, both assisted and personal, to practically deepen the theoretical aspects and the tools (specialized software) introduced, and in-depth seminars for the remaining hours. Everything will be supported by multimedia teaching material available online.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	**X**
Discussion of project paper	**X**
Other	

Each student will have to develop during the course 2 Homeworks, in addition to the final design work, including the use of the techniques and tools presented.

b) Evaluation methods:

During the discussion of the project work, theoretical insights will be requested relating to parts of the project itself, and everything will contribute to the final evaluation.

COURSE DETAILS

"AI SYSTEMS ENGINEERING"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ROBERTO PIETRANTUONO
PHONE: 0817683880
EMAIL: ROBERTO.PIETRANTUONO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

Algoritmi, Strutture Dati e Machine Learning

PREREQUISITES (IF APPLICABLE)

Ingegneria del Software

LEARNING GOALS

The course aims to provide the methodologies and tools necessary for the development and evolution of AI-based systems, with reference to the entire life cycle: design, development, verification and validation, evolution / maintenance, operation. The course illustrates the peculiarities of AI-based systems compared to traditional systems (such as the fundamental role of data), how these peculiarities impact on the various phases of the system life cycle, and what processes, models, languages, techniques, technologies and tools are required for the engineering of high quality systems. The contents will be exposed with reference to application areas of increasing diffusion, such as systems for autonomous driving of cars, driverless aircraft (*unmanned aerial systems*), (federations of) *bots* used in *customer service*, *social media*, virtual reality / metaverse. The use of different software tools to support the various phases of the life cycle, and "*domain-specific*" tools, such as simulators for a specific application domain, is foreseen.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

At the end of the course, the student must demonstrate familiarity with all phases of the life cycle of an AI-based system, from requirements analysis, to design, testing, evaluation and quality assurance. For each phase, the student will acquire knowledge and skills related to the processes, techniques, languages and tools necessary for engineering.

Ability to apply knowledge and understanding

At the end of the course, the student must demonstrate the ability to integrate artificial intelligence skills and solutions (and, in particular, *machine learning -ML*) in the engineering processes of AI-based systems. The training process is oriented to provide the skills to use the tools necessary to design, test, evaluate and improve the quality of AI-based systems.

PROGRAM-SYLLABUS

Part I. Introduction.

Introductory concepts. *Learning-enabled autonomous systems, cyber-physical systems, self-adaptive systems*. Application examples. Life cycle of intelligent systems. Requirements analysis, design, development, V&V, *quality assessment,* evolution/maintenance. Runtime monitoring, fault tolerance.

Part II. Analysis and Design.

System-level design. Modeling of system requirements. *Model-based System and Software Engineering.* Simulation, co-simulation. Modeling languages (SysML, UML). Domain-specific modeling languages (DSML) for *self-adaptiveness*. Agile approaches, DevOps, DevSecOps, ModDevOps.

Model-level (AI/ML component) design. ML references. The ML workflow. MLOps. Design, model development and operational phase. *Data Engineering pipeline; Model Engineering pipeline, deployment, monitoring, logging; Code Engineering pipeline*.

Part III. Test.

ML model testing, verification and validation. Deep Neural Networks **(**DNN) test. Black-box test, combinatorial test. White-box test, adequacy criteria. The problem of the oracle. *Multiple-implementation testing*. Pseudo-oracles. *Metamorphic* testing and *mutation testing*. *Operational testing* for DNS accuracy estimation. Generative Adversarial *Networks: Adversarial testing*. Bot testing (*social testing*). Notes on formal verification of DNN. AI system testing, integration testing, acceptance testing. *Model-based* testing. Compliance testing.

Part IV. Quality assurance.
Quality attributes of AI systems. Dependability, security, privacy. Interpretability, explainability. Performance analysis. Self-* attributes. Techniques of Fault *avoidance, Fault tolerance, Fault removal* in ML. Model-based, *measurement-based* quality estimation. Safety certification of AI systems. *Different redundancy, runtime monitoring.* Reliability estimation.

Design and test examples/case studies in application domains. Self-driving systems; driverless aircraft; bots used in *customer service*, social media, virtual reality/metaverse.

TEACHING MATERIALS

TEXTBOOK
Engineering Intelligent Systems - Systems Engineering and Design with Artificial Intelligence, 2022, Barclay R. Brown, Wiley. ISBN: 978-1-119-66563-2

TRANSPARENCIES FROM LESSONS AND EXERCISES available on the University professors website on the Microsoft Teams platform.

TEACHING METHODS

The teacher will use: a) lectures for about 70% of the total lesson hours, b) guided exercises for about 30% of the total lesson hours.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

The exam includes the development of a project paper and an oral test.

(b) Evaluation procedures:
The project paper and the oral exam are oriented to ascertain the knowledge and skills acquired in all phases of the life cycle of an AI-based system. The evaluation is therefore proportional to the degree of knowledge and ability to apply what has been learned at each stage.

COURSE DETAILS

"INFORMATION RETRIEVAL SYSTEMS"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ANTONIO M. RINALDI
PHONE: 0817683911
EMAIL: ANTONIOMARIA.RINALDI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide students with specialized notions to address and solve problems related to the Information Retrieval (IR) process. The course will present methods, models and techniques for searching for information and will describe technological and architectural aspects of IR systems. Methodologies for the representation of both textual and multimedia information through special descriptors will be presented, data mining, machine learning and deep learning techniques will be used for their analysis and BigData-based storage technologies in IR applications with particular reference to the semantic web, web of data and smart devices. Software tools will be used for the complete implementation of an Information Retrieval system and case studies on emerging applications will be presented.

EXPECTED LEARNING OUTCOMES

Knowledge and understanding

The training course aims to provide students with the knowledge and methodological tools necessary to analyze the problems related to the analysis of information to allow their retrieval. These tools will allow students to understand the main relationships between representing information, analyzing and managing it and to grasp the consequences in terms of effectiveness and efficiency in the context of the entire Retrieval process.

Ability to apply knowledge and understanding

The training course is aimed at transmitting the methodological and operational skills and tools necessary to concretely apply the knowledge related to the analysis of information in the Retrieval process for the development of techniques and the use of appropriate technologies.

PROGRAM-SYLLABUS

INTRODUCTION TO INFORMATION RETRIEVAL (IR) - The Information Retrieval Problem, Data retrieval and Information retrieval, Users, IR Systems, The WEB, Information Relevance, Search Interfaces, Navigation and Search, Query Specification, Results Display.

RETRIEVAL MODELS - Modeling and Ranking, Boolean Model, Vector Model, Probabilistic Model, Advanced Models for Information Retrieval.

EVALUATION OF THE RETRIEVAL PROCESS - Cranfield's Paradigm, Precision/Recall, F-measure, Other Measures for Evaluation, Document Collections.

RELEVANCE FEEDBACK AND QUERY EXPANSION - The User in the Reward Process, A Framework for Feedback Methods, Explicit Relevance Feedback, Implicit Relevance Feedback, Query Expansion.

DOCUMENTS AND QUERIES - Document Formats, Text Properties, Document Preprocessing, Thesauri, Ontologies, Text Mining, Multimedia Descriptors, Query Languages, Query Properties, Multimedia Queries.

TEXT CLASSIFICATION - Text Classification Characterization, Unsupervised Algorithms, Supervised Algorithms, Evaluation Metrics, Class Organization.

MULTIMEDIA RETRIEVAL - Definition of Multimedia, The Semantic Gap, Image Retrieval, Audio and Music Retrieval, Video Retrieval (outline), Multimodal Information Retrieval and Information Fusion, Artificial Intelligence Techniques, BigData and Deep Learning for Retrieval and Multimedia Classification.

INDEXING - The process of Indexing, Inverse Indexes, Search, Ranking, Construction, Structural Queries. WEB RETRIEVAL - The Web, The Structure of the Web Graph, Modelling the Web, Link Analysis, Search Engine Architectures, Search Engine Ranking, Semantic Web, Web of Data, Search Engine User Interaction.

WEB CRAWLING - Applications of a Web Crawler, Architecture of a Crawler, Scheduling Algorithms.

TEACHING MATERIALS

Textbook:

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, Modern Information Retrieval: The Concepts and Technology Behind Search 2ed, Addison Wesley, 2011.

Course slides.

TEACHING METHODS

Lectures, exercises, seminars (Artificial Intelligence, Big Data and Deep Learning Technologies for Advanced IR Applications), use of specialized software (Lucene, Apache Solr, LIRE, OPENCV)

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
Discussion of project paper	1
oral	1

The preparation will be evaluated through the discussion of a practical paper (description of the techniques and code development) proposed by the student and agreed with the teacher on a specific topic of the course. After the discussion of the paper there will be an oral exam on the theoretical / methodological part of the course. The tests contribute equally to the final grade.

(b) Evaluation procedures

COURSE DETAILS

"BIG DATA ENGINEERING"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: VINCENZO MOSCATO
PHONE: 0817683835
EMAIL: VINCENZO.MOSCATO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide in a specialized form the peculiar characteristics of the Architectures for Big Data Processing and Big Data Analytics, with reference to the design of large and complex data management systems, and to the implementation of the processes of modeling, acquisition, sharing, analysis and visualization of information present in Big Data. In particular, the most widespread technologies, frameworks and tools for the storage, processing and analysis of Big Data will be deepened, providing the student with all the knowledge necessary both for the development of practical applications for the application of Big Data to the so-called XInformatics, and for the understanding and management of the related technological infrastructures. Finally, the engineering issues related to the implementation of analytics on massive datasets for both batch and real-time applications will be treated, with reference to those that are the emerging applications of Big Data (eg Social Network Analysis, Cyber-Security, Smart Cities, etc.).

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course aims to provide students with specialized knowledge and all the methodological and technological tools for Big Data management, with particular reference to the characteristics of modern IT architectures for Big Data Processing and Big Data Analytics. In particular, these tools will allow students, depending on the type of application, on the one hand, to know how to define the most appropriate processes for modeling, acquisition, sharing, analysis and visualization of information related to Big Data, on the other to identify the hardware and software requirements of the supporting computer systems.

Ability to apply knowledge and understanding

The training course is aimed at transmitting the skills and methodological and operational tools necessary to concretely apply the knowledge related to the management of Big Data in real application contexts. In particular, the student must demonstrate to be able to design and implement in prototype form, exploiting the technological frameworks and existing tools, systems for Big Data Processing and Big Data Analitycs able to support different applications both batch and real-time.

PROGRAM-SYLLABUS

1. **Introduction to Big Data**: Definition and characteristics of a Big Data System.
2. **The Hadoop ecosystem**: HDFS, Hadoop, Yarn.
3. **Big Data analysis**: Pig, Hive, Giraph, Spark, GraphX. MLib.
4. **NoSQL databases**: Key-value - Column-family, Graph, Document Database systems. BASIC properties vs transactions. Theorem CAP. Notes on NewSQL Databases.
5. **Big Data Analytics (BDA):** BDA Lifecycle: knowledge discovery in database, data preparation, model planning, model building, data visualization.
6. **Real-time Big Data analysis**: Lambda and Kappa Architectures, Apache Flume, Apache Kafka, Spark Streaming.
7. **Emerging applications of Big Data**: Social Network Analysis, Cyber-Security, Smart-Cities.
8. **Cloud services for Big Data**: AWS, Microsoft Azure.

TEACHING MATERIALS

- Lecturer's notes, Lucidi, Scientific articles.
- Manuals of the various tools presented.
- Recommended textbooks:
    - "Big Data: Architecture, technologies and methods for the use of large databases",
    - A. Rezzani, APOGEO, 2013. "Business intelligence. Mathematical models and systems for decision-making",

- C. Vercelli, MacGraw-Hill Companies,2006 "Mining of Massive Datasets", J. Leskovec, A. Rajarman, J.D.Ullman, 2014 (on line book)

TEACHING METHODS

The teacher will use lectures for about 60% of the total hours, and in addition computer exercises, both assisted and personal, to practically deepen the theoretical aspects and the tools (specialized software) introduced, and in-depth seminars for the remaining hours. Everything will be supported by multimedia teaching material available online.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	X
Other	

In the case of a written test, the questions are (*)	Multiple choice	
	Free answer	
	Numerical exercises	

(*) You can answer multiple options

Each student will have to develop during the course 2 Homeworks, in addition to the final design work, including the use of the techniques and tools presented.

b) Evaluation methods:

During the discussion of the project work, theoretical insights will be requested relating to parts of the project itself, and everything will contribute to the final evaluation.

COURSE DETAILS

"DISTRIBUTED SYSTEMS AND IOT"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: STEFANO RUSSO
PHONE: 0817683832
EMAIL: STEFANO.RUSSO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide knowledge of advanced algorithms that solve classical problems in the engineering of software systems and distributed *embedded* systems (clock synchronization, consensus; global state; group communications; mutual exclusion; elections; transactions; fault tolerance; data consistency), as well as those used in *peer-to-peer* systems , in IoT systems and systems based on blockchain technology.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate: to understand the theoretical (non-technological) problems underlying distributed systems in reliable, critical, fault-tolerant and/or scalable applications; to know the theoretical limits as a function of temporal assumptions and failures; to know the algorithms for their resolution.

Ability to apply knowledge and understanding

The student must demonstrate to be able to abstract the non-technological problems of the engineering of a distributed system - be it software for network applications, or *embedded* in industrial applications - to be able to trace them back to known theoretical problems, and to be able to identify an adequate engineering approach to practical resolution.

PROGRAM-SYLLABUS

Characterization of distributed systems. Definitions. Architectural models. Fault model. Project requirements: performance, quality of service, scalability, fairness, safety, fault tolerance. Classification: synchronous, asynchronous, partially synchronous distributed systems. (0.10 CFU)

Time and synchronization in distributed systems. *Clock*, *skew*, *drift*, astronomical time , Coordinated Universal Time (UTC). Clock synchronization; Berkeley algorithm; Cristian's algorithm. Network Time Protocol (NTP). *Happened-before* relationship. Lamport logic clocks. Vector clocks. (0,5 CFU)

The global state of a distributed system. Substantial and inconsistent cuts. Chandy and Lamport *snapshot* algorithm. Snapshot reachability. Application examples. (0,4 CFU)

Consensus in distributed systems. The problem of consensus, of Byzantine generals, of interactive consistency. Dolev's algorithm. Paxos algorithm. *Failure detectors;* classification. *Lazy FD* algorithm. Consent with *failure detectors*. (1.25 CFU)

Group communications. Reliable and/or uncluttered multicasts. Algorithms for their implementation. (0,5 CFU)

Distributed coordination. Distributed mutual exclusion; algorithms: of the central server, ring, of Ricart and Agrawala, of Maekawa. Distributed elections. Algorithm: ring, bullying. Evaluation of algorithms. (0,5 CFU)

Distributed atomic actions. Transactions. ACID property. Data consistency issues , serializability, domino effect. Concurrency control: *Two-phase lock. Two-version lock*. Compound transactions. Distributed transactions, *two-phase commit*, concurrency control in distributed transactions. *Deadlock* distributed. Stable memory. (0.75 CFU)

Replication and fault tolerance in distributed systems. *Dependability*. The fault-error-failure chain. Active and passive replication. Adjudication strategies. The *state machine replication* approach. (0.25 CFU)

Data consistency across distributed systems. Replication and consistency. Tight, linearizable, sequential, causal, eventual, *processor consistency*, *slow memory*, client-centric models. CAP. theorem (0.5 CFU)

Blockchain and IoT. Blockchain: basic algorithms. *Smart contracts. Bitcoin*, *Ethereum, Hyperledger Fabric.* Internet of Things *(IoT).* Security aspects in IoT. Blockchain applications to IoT. (1 CFU)

Peer-to-peer systems. Motivations, requirements, applications, classification. Overlay networks, Overlay routing, Distributed Hash Table (DHT). Examples: Napster, Gnutella, BitTorrent, Chord, Pastry. (0.25 CFU)

TEACHING MATERIALS

Textbooks:

- Coulouris, Dollimore, Kindberg, "Distributed Systems: Concepts and Design – 5th ed.", Addison-Wesley, 2011;
- Kshemkalyani, Singhal, "Distributed Computing: Principles Algorithms and Systems", Cambridge Univ. Press, 2008.

Transparencies of the lessons (available on the teacher's website).

Scientific publications:

- L. Lamport, "Time, Clocks and the Ordering of Events in a Distributed System". Communications of the ACM, Vol. 21, No. 7, July 1978.
- L. Lamport, "Paxos Made Simple", 2001.
- F.B. Schneider, "Implementing Fault-Tolerant Services Using the State Machine Approach: A tutorial". ACM Computing Survey, Vol. 22, No. 4, December 1990.
- A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems". Proc. 18th IFIP/ACM Int. conf. on Distributed Systems Platforms (Middleware 2001), Nov. 2001.
- M. Crosby, Nachiappan, P. Pattanayak, S. Verma, V. Kalyanaraman, "BlockChain Technology", 2015, https://scet.berkeley.edu/wp-content/uploads/BlockchainPaper.pdf

TEACHING METHODS

Teaching is provided:
a) 80% with lectures;
b) 10% with seminars on emerging issues of blockchain and its applications to the Internet of Things;
c) for 10% with guided exercises in the classroom to practically deepen aspects related to logical and vector clocks and the realization of the algorithms treated.

The topics of the lectures and seminars are exposed with the help of detailed transparencies, made available to the student in the teaching material through the official website of the teacher.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	
other	

The exam is divided into an oral test only, consisting of three questions on as many problems / algorithms exposed to the course. At the student's choice, one of the three questions can be replaced by the presentation of the content of one of the scientific publications listed in the teaching material.

(b) Evaluation procedures:

COURSE DETAILS

"HIGH-PERFOMANCE AND QUANTUM COMPUTING"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ALESSANDRO CILARDO
PHONE: 0817683852
EMAIL: ALESSANDRO.CILARDO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to provide students with specialized notions related to thearchitectures of today's computers used for high performance computing, deepening concepts related to the internal structure of superscalar processors, and then extending the discussion to multi- and many-core computers. In this regard, the course also covers heterogeneous computing architectures, in particular GPU-based, and related programming models. The discussion of advanced computing architectures is then extended to emerging Quantum Computing technologies, introducing students to issues of design and system management and programming aspects.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course, based on theoretical and practical lessons, aims to provide the student with the knowledge related to the architectures of today's computers used for high performance computing (HPC), with reference to the different forms of parallelism offered to applications. The course therefore deepens the understanding of the internal structure of superscalar processors, and then extends the discussion to multi- and many-core computers. The first part of the program is also aimed at presenting heterogeneous computing architectures, in particular GPU-based, a paradigm established for the development of high-performance parallel applications. The second part of the course is aimed at deepening the opportunities offered by emerging Quantum Computing (QC) technologies in the perspective of high performance computing, addressing issues of design and system management and programming aspects.

Ability to apply knowledge and understanding

The training course is aimed at transmitting the methodological / operational skills and tools necessary to apply knowledge in terms of development of parallel applications, programming for advanced and emerging computing architectures, analysis of performance aspects. Specifically, the practical part of the course is dedicated to the presentation of programming models for GPUs, in particular CUDA and OpenCL. In addition, it provides insights into emerging Quantum Computing technologies, focusing on experimental QC platforms currently made available by international industrial companies. In this sense, the course includes the presentation of real case studies, partly developed in an interactive form with the students, with reference to both conventional parallel architectures and emerging QC scenarios.

PROGRAM-SYLLABUS

The course is organized in two types of contents, one theoretical and one applicative.

Theoretical part

- Superscalar architectures, out-of-order execution, hardware multi-threading, etc.
- Organizing Memory in Parallel Systems: Consistency and Consistency Issues
- System-level interconnects and on-chip networks
- Vector Extensions: General Concepts and Case Studies for Real Processors
- Graphics Processing Unit: architectural aspects and real case studies
- GPU Programming Models: Insights into Advanced Programming Concepts and Aspects
- Quantum Computing: technological aspects and computational models

Application part
For the laboratory, students will individually develop a paper that will include the vertical deepening of technological aspects on advanced or emerging computing architectures (QC), or alternatively the optimized development and presentation of a program on a GPU architecture or a vector extension.

TEACHING MATERIALS

- J. Hennessy, D. A. Patterson, *Computer Architecture: A Quantitative Approach*, 6th Edition, Morgan Kaufmann, 2019
- R.S. Sutor, *Dancing with Qubits: How quantum computing works and how it can change the world*, Packt Publishing, 2019
- NVIDIA, *CUDA C Programming Guide*, v. 11.1.0, online, NVIDIA 2020
- V. Silva, *Practical Quantum Computing for Developers*, Apress, 2018
- D. Kaeli, P. Mistry, *Heterogeneous Computing with OpenCL 2.0*, Morgan Kaufmann, 3rd edition, 2015
- D. J. Sorin, M. D. Hill, and D. A. Wood, *A Primer on Memory Consistency and Cache Coherence*, Morgan Claypool 2011
- N. E. Jerger, T. Krishna, and L.-S. Peh, *On-Chip Networks*, Morgan Claypool, 2nd edition, 2017
- Manuals and handouts provided during the course.

TEACHING METHODS

The teacher will use:
a) lectures for about 65% of the total hours,
b) exercises to deepen theoretical aspects for 8 hours,
c) laboratory to deepen the applied knowledge for 6 hours,
d) seminars to deepen specific themes for 2 hours.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	**X**
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

b) Evaluation methods:

COURSE DETAILS

"REAL-TIME SYSTEMS AND INDUSTRIAL APPLICATIONS"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: MARCELLO CINQUE
PHONE: 0817683874
EMAIL: MARCELLO.CINQUE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

REQUIRED PRELIMINARY COURSES (IF MENTIONED IN THE COURSE STRUCTURE

No

PREREQUISITES (IF APPLICABLE)

Basic knowledge on operating systems and programming, acquired during the bachelor's degree.

LEARNING GOALS

The course provides students with advanced notions related to real-time systems and their application in different industrial areas, with particular reference to mission and safety-critical systems. It provides the necessary skills for designing and developing real-time mixed-criticality systems, using operating systems and virtualization platforms for real-time embedded systems, including hybrid high-performance hardware architectures. The course addresses both the recommendations imposed by certification standards in different industrial contexts, such as automotive, railways and avionics, and research initiatives on related themes, such as the Industrial Internet of Things and Industry 4.0.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student needs to show to understand the theoretical problems at the base of real-time computing, to know the scheduling and resource management algorithms to address such problems. He/she has to be able to illustrate the virtualization principles and techniques that can be used to realize mixed-criticality systems while respecting isolation properties, to recognize the main industrial areas in which such systems are used, along with the constraints imposed by the normative.

Applying knowledge and understanding

The student needs to show to be able to apply the methodological instruments for the feasibility analysis and the dimensioning of real-time systems on single-core, multi-core and/or asymmetric/hybrid platforms; to be able to analyze the theoretical and practical problems concerning the use of operating systems and virtualization platforms in embedded and industrial environments, to be capable to design and implement complex real-time mixed-criticality systems, with reference to processes and technologies conformant to the main industrial standards.

COURSE CONTENT/SYLLABUS

Introduction to real-time systems. Fundamental properties. Hard and soft real-time. The safety concept, safety- critical systems. Scheduling of periodic tasks: cyclic executive, rate monotonic (RM), earliest deadline first (EDF). Aperiodic servers: polling server, deferrable server, sporadic server, total bandwidth server, hard constant bandwidth server. Resources management: non-preemptive protocol, highest locker priority, priority inheritance and priority ceiling. Feasibility tests based on upper bounds and on response time analysis.

Real-time kernel architectures. Real-time operating systems. Latency sources in computing platforms. Real-time executive. Monolithic kernels. Microkernels and L4 family. Dual kernels: RTAI and Xenomai examples. Preemptable kernels. The PREEMPT_RT patch for the Linux kernel.

Programming real-time tasks in Linux. The RT-POSIX standard. Programming cyclic periodic tasks. Fixed priority scheduling in Linux (SCHED_FIFO). Programming real-time applications on Xenomai. Real-time performance evaluation. Practical applications.

Real-time systems monitoring. Monitoring methodologies: embedded constraints and monitored constraints. Fault tolerance and timing failures management. Rule-based logging. Error propagation analysis.

Real-time multi-processing. Use of multiprocessing systems for real-time computing. Tasks models and theoretical limits. Symmetric and Asymmetric Multi-Processing. Partitioned scheduling of sporadic tasks with EDF and RM. Bin packing allocation. Global scheduling: the pfair algorithm, global EDF, Dhall effect, pfEDF, RM-light and RM-US. Feasibility bounds. Interference and isolation in multiprocessors due to memory hierarchies.

Mixed-criticality systems. The Vestal's model. Multi-criticality response time analysis. Audsley method. Hierarchical scheduling: open system architecture, fixed-priority hierarchical systems, hierarchical constant bandwidth server. The use

of hybrid asymmetric platforms and multi-processor systems on chip for the realization of mixed-criticality systems. OpenAMP: RemoteProc, RPMsg and examples of use on the Zync Ultrascale+ platform.

Real-time virtualization. Introduction to virtualization: CPU, memory and I/O virtualization. Virtualization issues in real-time systems. Types of hypervisors. Examples of real-time hypervisors: XEN with null and RTDS schedulers, Jailhouse and practical examples of use. Real-time cloud and real-time containers: architectural alternatives and examples of implementation/use in dual-kernel environments.

Industrial applications. The automotive context: the ISO 26262 standard, ASIL and development process, coding rules and MISRA-C, the concept of SEooC, the OSEK standard, the AUTOSAR standard, the CAN protocol, operating systems and hypervisors used in automotive. The avionics context: the DO 178B standard, the ARINC 653 standard, IMA architecture and APEX interface, real-time networking in avionics, operating systems and hypervisors used in avionics. The railways context: the CENELEC EN 50128 standard. The Industrial Internet of Things: terminology, architecture, notes on standardization initiatives and communication protocols.

READINGS/BIBLIOGRAPHY

- Textbooks:
  - Giorgio Buttazzo: "Hard real-time computing systems: Predictable Scheduling Algorithms and Applications", Third Edition, Springer, 2011.
  - S. Baruah, M. Bertogna, G. Buttazzo. "Multiprocessor Scheduling for Real- Time Systems", Springer, 2015
- Lessons' slides, reports and scientific articles available on the official course website.

TEACHING METHODS

Teacher will use a) lectures for approx. 75% of total hours, b) practical guided exercises in classroom to deepen the understanding and skills on real-time tasks programming and the use of hybrid virtualization architectures for about 20% of total hours, and c) seminars on specific themes, such as industrial standards, for about 5% of total hours.
Lectures and seminars are explained with the help of detailed slides, which will be provided to students through the official course website.

EXAMINATION/EVALUATION CRITERIA

e) Exam type:

Exam type	
written and oral	
only written	
only oral	X
Project discussion	X
other	

The exam is articulated in a single oral session, which comprehend the presentation of a project assigned during the course and three questions on problems, algorithms and theoretical/technological solutions among the ones explained at the course. The project can be developed in teams of maximum three students.

COURSE DETAILS

"EMBEDDED SYSTEMS"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ALESSANDRO CILARDO
PHONE: 0817683852
EMAIL: ALESSANDRO.CILARDO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to provide students with specialized notions related to modern *Embedded Systems* (dedicated computer systems expressly designed and integrated in "embedded" form in specific products, often bound to meet certain real-time and performance requirements, as well as requirements on consumption, size, reliability and safety) such as those widely used in industrial systems (transport: automotive and railway; energy; mechanical control systems) or consumer control systems (telephony, entertainment, multimedia processing) or, in general, Internet of Things (IoT), e-health, robotics and Artificial Intelligence systems.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course, based on theoretical and practical lessons, is aimed at providing the student with the knowledge related to embedded architectures, with emphasis on technological and methodological aspects, as well as the understanding of design flows for System-on-Chip (SoC) and Multi-Processor System-on-Chip (MPSoC) systems. The course will present, among others, ARM processor SoC systems on STMicroelectronics STM32F4 boards and other processors for embedded applications synthesized on FPGAs. In the laboratory lessons, knowledge of some IDE environments for design is also developed, consisting of compilers of HDL languages, such as VHDL and Verilog, debuggers, simulators and tools for technology mapping.

Ability to apply knowledge and understanding

The training course is aimed at transmitting the methodological / operational skills and tools necessary to apply the knowledge in terms of engineering design of embedded systems, based on different System-on-Chip (SoC), Multi-Processor System-on-Chip (MPSoC) and special (DSP, dedicated hardware) architectures, also synthesized with FPGA components already studied in previous courses of the master's degree course. The design will make use of development methodologies and professional IDE environments widely used in the industrial world. The application part, in particular, involves the development of a classroom project.

PROGRAM-SYLLABUS

The course is organized in three main parts: a theoretical, a technological and an applicative one.

Theoretical part

- Special and general architectures of Embedded Systems and their layered structuring.
- Hardware architectures based on the integration of commercial hardware (processors, peripherals, multiprocessors, multicomputers, DSP, system on chip, etc.), specific hardware (designed using HDL and automatic synthesis technologies) and devices for connection with actuators and field sensors.
- Architectural characteristics of processors and systems for System-on-Chip (SoC) and System-on-Chip (MPSoC) Multi-Processor applications.
- Basic software, operating systems and application software for dedicated systems with or without real-time constraints.
- General information on sensors and actuators: operating principles, type, interfaces and examples of use.
- Sensor networks: architecture and service software.
- Partitioning logics of a dedicated system between functional units realized in hardware and / or software of embedded SoC and MPSoC systems.

- Processing systems for industrial applications critical for cost, reliability, time, safety, consumption and footprint.
- Development cycle: methodologies and tools for synthesis and co-design.
- Development of an STMicroelectronics STM32xxx series embedded board system with ARM processors.

Technological part

- Hardware/software layers in which an embedded system is structured: HAL level, middleware, RTOS operating system and application, even with real-time requirements, I/O subsystem characterized by the most common I/O devices, sensors and actuators and their networks.
- I/O devices can be summarized as IP *cores* (UART/USART, PIO, I2C, SPI, USB, etc.) on FPGAs.
- Basic components: memory subsystems, interconnect architecture. High-performance buses and interconnection systems: AMBA and AXI buses.
- Architectures of the most popular processors for embedded systems: the ARM family; native processors of the FPGA family (Xilinx's MicroBlaze); other processors that can be synthesized as IP cores on FPGAs (low-end 8, 16, 32-bit processors of type CISC and RISC, Java machine); RISC-V open architecture.
- System-on-Chip architectures, STM32Fxx family of systems.
- Architecture of parallel multiprocessor systems and interconnection networks.
- Architecture of Multicore Systems and Multi-Processor System-on-Chip.
- Hybrid programmable components composed of SoC or MPSoC (with general-purpose RISC/CISC and/or special processors such as Digital Signal Processors, DSPs) and FPGAs (to synthesize special machines).
- FPGA-multicore hybrid devices: the Zynq family.
- Architecture of Digilent Zybo Zynq and ZedBoard Zynq Boards
- IP-Core libraries, "intellectual Property Core", specially developed or present in public and commercial libraries.

Application part

For the laboratory, the students, organized in working groups, will develop a concrete project of embedded system, divided into subsystems, starting from the functional specifications assigned to them. The design will use development methodologies and supporting IDE environments, concretely used in the industrial field. In particular, reference will be made to examples of commercial "on board" systems made available to students, with processors of the ARM family and FPGAs of the Xilinx families.

TEACHING MATERIALS

- G. Conte, A. Mazzeo, N. Mazzocca, P. Prinetto, "Architettura dei calculators", Cittastudi edizioni, 2015.
- F.Vahid, T.Givaris, "Embedded System Design - A unified Hardware/Software Introduction", Wiley, 2002.
- K. Papovic, F. Rousseau, A. A. Jerraya, M. Wolf, "Embedded Software Design and Programming of Multiprocessor System-on-Chip", Springer, 2010.
- M. Domeika, Software Development for Embedded Multi-core Systems a Practical Guide Using Embedded Intel ® Architecture
- "Automotive Embedded Systems Handbook, Industrial Information Technology Series CRC Press, 2009, N. Navet, F. Simonot Lion eds.
- K. Yaghmour, J. Masters, G. Ben-Yossef, and P. Gerum, "Building Embedded Linux Systems", O'Reilly.
- Wayne Wolf, "Computers as components", Morgan Kaufman, 2000. Planned 2nd ed
- P. Marwedel,"Embedded System Design", Kluwer academic publishers, ISBN: 1-4020-7690-8, November 2003
- J. Sauermann, M. Thelen, "Realtime Operating Systems Concepts and Implementation of Microkernels for Embedded Systems",
- Manuals and datasheets of the various devices used and supplied during the course.

TEACHING METHODS

The teacher will use:
a) lectures for about 55% of the total hours,
b) exercises to deepen theoretical aspects for 8 hours,
c) laboratory to deepen the applied knowledge for 10 hours,
d) seminars to deepen specific topics for 4 hours.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	X
other	

In the case of a written test, the questions are (*)	Multiple choice	
	Free answer	
	Numerical exercises	

b) Evaluation methods:

COURSE DETAILS

"WEB AND REAL COMMUNICATION SYSTEMS

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: SIMON PIETRO ROMANO
PHONE: 0817683823
EMAIL: SIMONPIETRO.ROMANO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

REQUIRED PRELIMINARY COURSES (IF MENTIONED IN THE COURSE STRUCTURE "ORDINAMENTO")

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the class is to provide students with advanced notions in the field of both web-based and real-time communication. Students will become familiar with the design and development of complex communication systems, by focusing both on the client-side and on the server-side. They will become acquainted with the networking protocols that form the basis of web-based and real-time communication, as well as learn how to use the standard Application Programming Interfaces (APIs) laying on top of them. This is both a programming class and a networking class. All of the networking protocols and solutions will be first studied in detail and then put into practice through the design and implementation of proof-of-concept prototypes representing real-world application scenarios.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

At the end of the class, students will be able to demonstrate advanced knowledge in the field of web-based and real-time communication. They will have a clear understanding of the protocols and APIs representing the state of the art in both fields. Step by step, the class will bring them towards an integrated approach, whereby web-based and real-time communication eventually co-exist in a single, unified scenario adhering to the latest standards issued by both the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C).
They will be ready to enter the professional arena and become part of cutting-edge development teams, by actively contributing to the design and the implementation of advanced communication systems. Successful completion of the class will allow them to effectively apply the acquired knowledge to a number of real-world scenarios requiring advanced engineering capabilities, by putting together their networking competences and their advanced programming skills.

Knowledge and understanding

Students need to show ability to know and understand problems related to both the communication and the programming issues associated with the design and development of complex interoperating distributed systems.
They need to elaborate arguments related to the relationship between communication protocols, synchronous and asynchronous interaction, client-side, server-side and peer-to-peer programming, by embracing an engineering approach and looking at them in an integrated fashion. The class provides students with advanced knowledge in the field of web-based real-time communication, by also illustrating how to leverage both methodological and practical tools in order to design and implement effective, interoperable, scalable and secure real-time multimedia communication systems that are compliant with state-of-the-art standard protocols and APIs. Such tools will allow students to grasp the causal connections among network-based communication and event-based programming, as well as understand the implications of the adoption of an agile design and development paradigm for the realization of advanced communication systems.

Applying knowledge and understanding

Students need to show ability to design complex systems involving distributed components that exchange multiple media in a real-time fashion. They will have to demonstrate a clear understanding of the main networking protocols offering support to such systems for all what concerns their communication requirements. They will have to demonstrate advanced programming skills, with special reference to the use of the standard APIs that are offered to programmers both on the client-side and on the server-side. The course delivers skills and tools needed to apply knowledge in practice, favoring the ability to use a methodological approach that properly integrates different technologies (as well as different programming languages) in a unified framework allowing to effectively look after the many facets of a complex communication system.

COURSE CONTENT/SYLLABUS

The class will proceed step-wise towards the final goal of designing and implementing integrated systems putting together the most up-to-date solutions in the fields of web-based and real-time communication. Web-based communication protocols and technologies will be first introduced, by focusing both on the communication and on the programming aspects. Server-side web application development will be analyzed in detail, with the help of real-world examples associated with state-of-the-art frameworks and programming languages. Client-side web programming through the JavaScript language will be discussed, with a focus on both the synchronous and the asynchronous paradigm. The XML (eXtensible Markup Language) language and APIs will be introduced and analyzed in depth. The focus will then move to real-time communication, by looking at both the data layer and the signaling layer. State-of-the-art streaming technologies will be discussed in detail and comparatively analyzed. Network reachability and NAT (Network Address Translation) traversal issues will be presented, together with the standard protocols that have been designed in order to effectively deal with them. A detailed overview of the most up-to-date Instant Messaging applications and related standard protocols will be presented. Finally, the class will show how to put things all together in an integrated framework that leverages both the IETF RtcWeb and the W3C WebRTC standard protocols and APIs in order to realize next-generation web-enabled real-time communication scenarios.

Syllabus:

- Web-based communication basics: HTTP protocol deep-dive
- Server-side programming basics: Common Gateway Interface (CGI)
- Server-side programming in Java through Java servlets
- Server-side programming in Python through the Flask framework
- Server-side programming in Java through the Spring Boot framework
- Client-side programming with JavaScript
- Asynchronous JavaScript and XML (AJAX) and the Fetch API
- The XML language
- XML programming
- Server-side programming in JavaScript with Node.JS
- Web-based interaction through WebSockets
- The Real-time Transport Protocol (RTP)
- Voice over IP (VoIP) applications and the Session Initiation Protocol (SIP)
- Streaming protocols and technologies:
    - Real Time Streaming Protocol (RTSP)
    - MPEG-DASH (Dynamic Adaptive Streaming over HTTP)
    - Real Time Messaging Protocol (RTMP)
    - Peer-to-peer approaches
- NAT-traversal protocols:
    - Session Traversal Utilities for NAT (STUN)
    - Traversal Using Relays Around NAT (TURN)
    - Interactive Connectivity Establishment (ICE)
- Instant Messaging protocols and technologies:
    - Internet Relay Chat (IRC)
    - Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions (SIP/SIMPLE)
    - eXtensible Messaging and Presence Protocol (XMPP)
    - WebRTC-enabled IM through data channels
    - Web-socket enabled IM
    - Message Queuing Telemetry Transport (MQTT) enabled IM
    - Advanced Message Queuing Protocol (AMQP) enabled IM
    - Simple Text Orientated Messaging Protocol (STOMP) enabled IM
    - The RabbitMQ library
    - Secure, decentralized real-time communication through Matrix
- User interface design and implementation through React.js
- Using native components instead of web components with React Native

- WebRTC: the new frontier of real-time communications in the web
- The Janus WebRTC media server and gateway

READINGS/BIBLIOGRAPHY

1. *Official references like, e.g., Requests For Comments (RFC), available at: https://www.ietf.org*
2. *"Real-Time Communication with WebRTC", Salvatore Loreto, Simon Pietro Romano, Released May 2014, Publisher: O'Reilly Media, Inc. ISBN: 9781449371876*
3. *Slides and additional materials provided by the teacher and made available on the official sites associated with the class*

TEACHING METHODS

Teacher will use a hands-on approach for the entire duration of the class. All of the course topics will be both presented in theory and further analyzed through practical examples.
All of the students will have to work on a practical project, either individually or in groups, focused on an in-depth study of one or more of the topics addressed during the class.

EXAMINATION/EVALUATION CRITERIA

a) Exam type:

Exam type	
written and oral	
only written	
only oral	X
project discussion	X
other	

In case of a written exam, questions refer to: (*)	**Multiple choice answers**	
	Open answers	
	Numerical exercises	

(*) multiple options are possible

b) Evaluation pattern:

- 35% of the final mark will depend on an evaluation of the practical project developed by the student (either individually or in group). The project itself will have to be delivered (with a fully-fledged documentation, including source code, if applicable) to the teacher at least seven days in advance of the oral examination date;
- 65% of the final mark will depend on the results of the oral interview.

COURSE DETAILS

"DECENTRALIZED APPLICATIONS AND BLOCKCHAIN"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: GIUSEPPE ACETO
PHONE: 0817683851
EMAIL: GIUSEPPE.ACETO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

REQUIRED PRELIMINARY COURSES (IF MENTIONED IN THE COURSE STRUCTURE "ORDINAMENTO")

No

PREREQUISITES (IF APPLICABLE)

Computer Networking, Computer Programming

LEARNING GOALS

The course aims to provide the skills for understanding the functioning of the main blockchain-based platforms (both public and permissioned) and for the development of smart contracts and decentralized applications (DApps).

The course also intends to provide the methodological tools to analyze and evaluate the set of technologies available for the implementation of DApps, to guide their design, development and adoption in different application contexts.

The course includes practical and laboratory classes functional to the development of a course project.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems arising and solutions available relating to the design and development of decentralized applications (DApps), with particular reference to blockchain-based architectures, the problem of consensus, and tools related to blockchains.
The student must also demonstrate knowledge and understanding of the concept of smart contracts, and the main languages for the programmability of blockchain-based architectures.

Applying knowledge and understanding

The student must demonstrate being able to design DApps taking into account the application context, the available technologies and their characteristics. They must also be able to develop simple DApps by effectively using the knowledge acquired regarding the programmability of blockchain-based architectures.

COURSE CONTENT/SYLLABUS

Part I - Background

Introduction and contextualization of cryptocurrencies, blockchain, smart contracts and decentralized applications.
Introduction to distributed systems, decentralized systems and the consensus problem. Distributed ledger technologies.
Fundamentals of cryptography: public key algorithms, digital signature, hashing, validation.
Consensus protocols: Proof of Work, Proof of Stake, Proof of Authority, other protocols.
Smart Contracts and Tokens.
Types of tokens, fungible and non-fungible (NFT).
Trust and communities: hard fork and soft fork.

Part II - Technologies

Blockchain Technologies: Bitcoin, Ethereum, IOTA and other major blockchains.
Scalability and consensus protocols; Layer 2 solutions: Lighting Network, Polygon.
Tools for using blockchain (wallet, chain explorer, gateway and API), off-chain systems, IPFS.
Lab #1: Use of test chains of selected blockchains

Part III - Programmability

Decentralized Applications on Blockchain (DApp).
DApp prototype development (smart contract + interface + off-chain systems).
Test-driven development of DApp.
DApps models and emerging standards.
Lab #2: Prototype implementation of DApp

Part IV - Challenges and perspectives
Security aspects related to DApps and smart contracts.
Network Security and Blockchain: Blockchain Threats and Blockchain-Based Solutions.
Blockchain privacy and blockchain interoperability (cross-chain technologies).
Performance measurement of blockchain-based technologies.
Integration of Artificial Intelligence and blockchain.
Applications of blockchain-based technologies in SDN, IoT, 5G/6G, eHealth contexts.
Decentralized Autonomous Organization (DAO).
Challenges and opportunities for the adoption and development of Blockchain and DApps.

READINGS/BIBLIOGRAPHY

- Lecture slides and notes
- Further teaching material provided by the teacher
- *Dannen, Chris. Introducing Ethereum and solidity. Vol. 1. Berkeley: Apress, 2017.*

TEACHING METHODS

The course will consist of:
a) lectures for approx. 70% of total hours;
b) laboratories and/or practical lessons for the remaining 30% of total hours.

EXAMINATION/EVALUATION CRITERIA

a) Exam type:

Exam type	
written and oral	
only written	
only oral	X
project discussion	X
other	

In case of a written exam, questions refer to: (*)	**Multiple choice answers**	
	Open answers	
	Numerical exercises	

(*) multiple options are possible

The oral exam will include the presentation and discussion of a course project previously prepared by the student.

b) Evaluation pattern:

COURSE DETAILS

"WIRELESS NETWORKS AND IOT TECHNOLOGIES"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: STEFANO AVALLONE
PHONE: 0817683902
EMAIL: STEFANO.AVALLONE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to impart an in-depth knowledge of the main technologies used in wireless networks, both high-performance (Wi-Fi) and characterized by constraints on the power consumption of devices (LoraWAN, ZigBee), as required for example by the Internet of Things (IoT) paradigm. This aim is pursued through the analysis of the main problems faced by the wireless technologies considered and the presentation of the most recent solutions proposed by international standardization bodies. The course is mainly focused on issues related to vehicle access, routing and support for applications in wireless networks. The main training objectives are: knowledge of the main algorithms distributed for access to wireless means; the acquisition of the main methodologies for the analysis of the performance of wireless access techniques; knowledge of security issues in wireless networks; understanding the issues arising from the need to reduce the power consumption of devices; knowledge of protocols for supporting applications in IoT networks; the ability to use tools for simulating wireless networks.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

Students must demonstrate knowledge of the main wireless local area network technologies and the main differences between them. Students must understand the issues related to access to shared wireless media and understand the design choices behind the different solutions proposed by the different technologies. The training course aims to provide students with the basic knowledge and methodological tools necessary to analyze the performance of wireless local area networks. Students must demonstrate knowledge of tools for capturing real network traffic and tools for running network simulations.

Ability to apply knowledge and understanding

Students must be able to identify the most suitable wireless local area network technology for a given application context, based on the characteristics of the devices and the environment in which they operate and on the basis of the application requirements imposed. Students must be able to select the techniques and mechanisms made available by a wireless network technology in order to achieve certain objectives. Students must be able to analyze the performance of the different techniques proposed by wireless local area network technologies also through the use of network simulation tools.

PROGRAM-SYLLABUS

WLAN networks: architectures, component definition, management and control procedures. WLANs: outline of the physical layer (IEEE 802.11a/b/g standards). MAC level: DCF and PCF. Security in WLANs: authentication mechanisms (WEP, PSK, 802.1x, SAE) and encryption algorithms (WEP, TKIP, CCMP, GCMP). Quality of Service Support Extensions: IEEE 802.11e standard and EDCA and HCCA medium access functions. Methodologies for evaluating the performance of IEEE 802.11-based WLANs. Evolution of WLANs: amendments 802.11n, 802.11ac and 802.11ax. Ad-hoc wireless networks: application scenarios, issues, reactive and proactive routing protocols. Mesh wireless networks: application scenarios, management and control procedures, protocols for path selection, interoperability with other LAN network segments. Wireless communication technologies for the Internet of Things (IoT): LoraWAN and Zigbee (IEEE 802.15.4 standard). Application protocols for IoT: MQTT and CoAP. Tools and methods for simulating wireless networks.

TEACHING MATERIALS

Recommended textbook:
Stefano Avallone, "Protocols for Mobile Networks". McGraw-Hill Italy. ISBN: 978-88-386-7414-3

Other useful teaching material:
Slides of the lessons, scientific articles and selected standards.

TEACHING METHODS

Teaching will be delivered mainly through lectures. About 6 hours will be dedicated to the illustration of the ns-3 network simulator, which will be used as a tool for performing network simulations. Documentation for the ns-3 simulator is available online on the simulator website.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

The paper consists in the realization of a program for the ns-3 network simulator that has as its objective the evaluation of one or more techniques used by the wireless network technologies covered by the course. The paper affects a maximum of 3 points on the overall evaluation.

COURSE DETAILS

"CLOUD PLATFORMS AND INFRASTRUCTURE-AS-CODE"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ROBERTO CANONICO
PHONE: 0817683831
EMAIL: ROBERTO.CANONICO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

Nobody

PREREQUISITES (IF APPLICABLE)

Sistemi Operativi, Reti di Calcolatori, Programmazione, Cloud and Network Infrastructures

LEARNING GOALS

The aim of the course is to impart an in-depth knowledge of the main methodologies and techniques used in Cloud Computing contexts (public, private and hybrid) for the dynamic sizing, configuration and management of virtualized infrastructures. In particular, the course deals with:
- the engineering aspects of the design and construction of a modern datacenter and the main IT technologies used in this specific context;
- the peculiar architectural and protocol solutions that characterize networking in the context of datacenters;
- the main virtualization techniques currently used for the various components of a Cloud-based IT system and how these are used for the creation of scalable, elastic, flexible and reconfigurable systems through the deployment and orchestration of VMs, containers and serverless components;
- automation techniques that, through the DevOps paradigm, allow to automate the procedures for commissioning, configuration and management of cloud and network systems.

The course also includes a laboratory and exercise part functional to the development of a paper.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student will have to demonstrate that he has understood the fundamental characteristics of the virtualization techniques used in cloud systems and how they must be managed for the construction and commissioning of scalable systems.

Ability to apply knowledge and understanding

The student will have to demonstrate that he has acquired the ability to apply the techniques learned for the resolution of typical problems of designing scalable applications in the cloud and for the configuration of the infrastructural components of a datacenter that hosts cloud applications.

PROGRAM-SYLLABUS

Part I – Cloud Data Center engineering

Datacenter architecture and organization. General architecture and organization of a datacenter. Main datacenter facilities. Rack layouts. Rackable servers. Blade servers. ANSI/TIA-942 standard. Data Center Infrastructure Management (DCIM).

Data center power management. Green datacenters and green cloud computing. Cooling solutions for datacenters. Power Usage Effectiveness (PUE).

Datacenter networking. Transmission media for datacenter links. UTP cabling. Optical fibers. Optical transceivers. Access layer organization: Top-of-Rack vs. End-of-Row. Leaf-Spine datacenter networks. Multipath in datacenter networks. ECMP and TRILL. Flowlets. TCP Incast and TCP variants for datacenter networks.

Storage networking technologies. Sans. Fiber Channel. Fibre Channel over Ethernet (FcoE). iSCSI.

Datacenters for HPC applications. Infiniband.

Virtualization technologies. The virtualization concept. Different approaches to IT resource virtualization. Type-1 and Type-2 hypervisors. VM networking.

Container-based virtualization. Docker. Docker Image Layers. Container image repositories. Docker networking.

Virtual routers. Virtualized network functions. Network Function Virtualization (NFV). Virtual Network Functions chaining and orchestration.

Multi-tenancy in cloud networking. IEEE 802.1ad Q-in-Q. Overlay networking. NVGRE. VxLAN.

Lab #1: Docker. Creation of Docker containers. Multi-stage docker build.

Part II - Cloud Platforms
Modern application design in public cloud contexts.
Private Cloud platforms. OpenStack. OpenStack general architecture and fundamental services.
Resource management and configuration in private cloud infrastructures.
A comparison of IaaS and PaaS offerings by main Cloud Service Providers.
The AWS, Azure and Google public cloud offerings.
Identity management in the Cloud. Cloud APIs and interoperability.
Microservices.
Serverless computing. AWS Lambda.
Cloud edge computing.
Lab #2: OpenStack.
Labs #3, #4: A practical introduction to public cloud offerings.

Part III - Cloud and network automation
Orchestration of Docker containers. Kubernetes.
Network automation. NETCONF and RESTCONF. YANG data models.
Infrastructure as Code paradigm.
DevOps and Cloud automation in private and public cloud platforms. Basic DevOps concepts. CI/CD.
Frameworks for automatic deployment and configuration of cloud components. Ansible. YAML. Terraform.
Lab #5: Use of YAML for automatic deployment of cloud components.

TEACHING MATERIALS

- Lecture notes and slides
- Other teaching material made available by the teacher
- *Cloud Native DevOps with Kubernetes,* 2nd Edition, Justin Domingus, John Arundel. O'Reilly Media, 2022
- *Cloud Native Data Center Networking: Architecture, Protocols, and Tools (1st Edition).* Dinesh G. Dutt. O'Reilly, 2019
- *Network Programmability and Automation.* Jason Edelman, Scott S. Lowe, Matt Oswalt. O'Reilly, 2018

TEACHING METHODS

The course will consist of: a) lectures for 80% of the total hours; b) exercises for the remaining 20% of the total hours.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

The exam will consist of an oral exam during which a project previously written by the student will also be discussed.

(b) Evaluation procedures:

COURSE DETAILS

"SYSTEMS SECURITY"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: VALENTINA CASOLA
PHONE: 0817683907
EMAIL: VALENTINA.CASOLA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Programming knowledge; knowledge of the principles of software engineering.

LEARNING GOALS

The course aims to provide a methodological and technological approach for the design of secure systems. The course aims to analyze the standard design techniques with reference to the development and use of the main security mechanisms, including: authentication and access control mechanisms, cryptographic security mechanisms, mechanisms for the protection of communications and distributed systems. The main elements for the analysis of risks and threats applicable to a system to guide the design phases and the main techniques of assessment and testing of the security of the systems are also presented.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems related to the design of secure systems, with reference to the standard analysis and design methodologies presented during the course, and considering the specific constraints deriving from the technologies used. They must also demonstrate an understanding of the key features of different safety mechanisms and identify the most appropriate controls to meet specific design requirements.

Ability to apply knowledge and understanding

The student must demonstrate to be able to perform the entire cycle of analysis, design and development of a secure system, from the risk and threat analysis phase to the identification of the most appropriate control mechanisms, to their implementation and correct configuration, up to the final testing of the system security, using widely used development tools and environments.

PROGRAM-SYLLABUS

Course Introduction: Basic terminology, Overview of system security, Policy/mechanism separation, Security requirements.

Fundamentals of cryptography: Symmetric cryptosystems: Block Cipher (DES, Skipjack....), Asymmetric cryptosystems: RSA, ECC; Key Management and distribution; Digital signature, Hash functions, Smart Card security; Public Key Infrastructure: PKCS Standards, X. 509 Certificates, Certificate Policies and Cross Certification; Java Cryptography Architecture; Digital Signature and PEC in the Italian law.

Identification and Authentication mechanism : Authentication mechanisms, Authentication protocols, Single Sign On, Kerberos, Identity Federation, OAuth, SAML, IAM (Identity and Access Management) Systems; credential management systems (Vault).

Access Control mechanism: Access Control models: Discretionary and Mandatory Access Control Models (DAC, MAC), Role based Access Control Models (RBAC), Other models: Attribute based Access Control (ABAC), Role hierarchy management, Conflict management; Access Control frameworks: XACML, Keycloack, Authentication and Authorization services.

System and Communication Protection mechanism: Attack taxonomy, Firewalls, Gateways, Intrusion Detection systems; Network segmentation and demilitarized zone (DMZ), Monitoring mechanisms; Auditing and Logging mechanisms.

Application and Network Security protocols: SSL, PGP, SMIME, VPN, IPv6.

Design of Secure Systems: standard risk-based development approach (NIST, ISO), Secure SDL methodologies, threats and vulnerabilities analysis, risk analysis, security controls identification techniques, security assessment, static and dynamic security testing techniques. Design trade-offs: Security and Performances. Case studies: Web application security, Security in hw and embedded devices (IoT security, WSN security, FPGA security.....), Cloud Security.

TEACHING MATERIALS

Textbook: William Stallings – Computer Security, Principles and Practice 3rd Ed - Prentice Hall.
Handouts and presentations provided by the teacher related to theoretical and applicative topics.
Manuals and reference standards of the safety mechanisms and methodologies used.
Code relating to classroom exercises.

TEACHING METHODS

The course includes about 70% of lectures in which the theoretical topics are addressed, while the remaining 30% is reserved for practical lessons and exercises concerning the design, implementation and evaluation of security mechanisms studied.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	x
Discussion of project paper	x
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

(*) You can answer multiple options

The assessment of learning includes an oral exam aimed at verifying the understanding of the theoretical concepts of the course and the discussion of a paper.

COURSE DETAILS

"NETWORK SECURITY"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: SIMON PIETRO ROMANO
PHONE: 0817683823
EMAIL: SIMONPIETRO.ROMANO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II):
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide students with advanced notions in the field of network security. Students will become familiar with the most well-known security mechanisms, as well as with the techniques of mitigation of cyber attacks, focusing on the solutions available at the various levels of the network protocol stack, from the physical layer to the application layer.
The course leverages some of the concepts covered in the course of "Secure Systems Design", with particular reference to symmetric cryptography, message confidentiality, public key cryptography and authentication. On the other hand, it introduces some of the topics that form the core of the "Software Security" course, such as, for example, buffer overflow attacks and fuzzing techniques.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

At the end of the course, students will be able to demonstrate advanced knowledge in the field of network security. They will have a clear idea of the vulnerability-threat-attack chain and will be able to design effective defense techniques. They will know how to secure a critical network infrastructure. They will be ready to enter the labor market as members of experienced cybersecurity teams, actively contributing to both the detection of network attacks, as well as those of reaction and implementation of the necessary corrective actions. The successful completion of the course will allow them to bring the knowledge gained to the field, playing a leading role within the so-called Security Operation Centers (SOC) and / or teams of professionals specialized in computer security.

Knowledge and understanding

Students will have to show that they can understand and deepen the problems related to the effective protection of a network architecture from cyber attacks. They will have to deepen topics related to the relationships between security at the level of the single node, the Operating System, the software and the network, framing everything in an integrated vision that makes use of a holistic approach. The course provides students with advanced knowledge in the field of cybersecurity, illustrating how to leverage both methodological and practical tools, in order to discover vulnerabilities, detect attacks, analyze interaction paradigms and behavioral patterns of users of a network, design and implement adequate defense measures, with both a "reactive" and "proactive" approach. These tools will allow students to identify the causal connections between the concepts of vulnerability, threat and attack, as well as to understand the implications deriving from the use of an offensive approach to improving the overall level of security of a network infrastructure.

Ability to apply knowledge and understanding

Students will have to show the ability to make decisions, evaluating the consequences, starting from the available information (real-time traffic, recorded traffic traces, log files, registry audits, application source code, etc.) in order to effectively manage the problems related to the presence of a possible distributed attack on the network. They will also have to demonstrate advanced knowledge of the techniques and tools that can be used to prevent an attack, detect it in real time, mitigate its effects and counteract its presence through the use of active remediation mechanisms. The course provides the skills and tools necessary to apply the acquired knowledge in practical scenarios, favoring the ability to employ a methodological approach that appropriately integrates different types of countermeasures (possibly available at different levels of the network protocol stack), within a homogeneous context that allows to take into account the multiple facets of a security attack.

PROGRAM-SYLLABUS

The main security properties of a computer system will be introduced and discussed in detail. Approaches to improve security at the various layers of the network protocol stack will then be presented and analyzed. The course adopts an "offensive security" approach. The concepts related to the so-called vulnerability-threat-attack chain will be illustrated. You will learn techniques for preparing for a cyberattack, such as *footprinting*, *scanning*, and *enumeration*. The final stage of an attack, known as *exploitation*, will eventually be presented. Topics such as firewalling, intrusion detection, malicious

software analysis, and distributed denial of service (DDoS) protection will be introduced. The main techniques of so-called ethical hacking will finally be presented and analyzed in detail.

Syllabus:

- Network security: principles and architecture
    - Functional safety requirements
    - threats, attacks, countermeasures
- Wireless Network Security
- Network-level security
    - the IPsec protocol suite
- Transport level security
    - Secure Socket Layer (SSL), Transport Layer Security (TLS), Secure Shell (SSH)
- Application-level security:
    - Email
    - Web
        - HTTPS
        - Web Real Time Communications (WebRTC) security architecture
- Cloud Computing and Security (outline)
- Introduction to malicious software
    - taxonomy
        - Viruses, worms, Trojans, rootkits, etc.
    - Advanced Persistent Threats (APTs)
    - Countermeasures
- Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks
- Intrusion Detection Systems (IDS)
    - "Host-based", "network-based" and hybrid techniques
- Firewall and Intrusion Prevention Systems (IPS)
- Hacking in IP networks
    - Preliminary stages of an attack:
        - Footprinting, scanning, enumeration
    - Oriented attack techniques:
        - end-systems and servers;
        - To the structure:
            - Voice over IP (VoIP) networks
            - Wireless Networks
            - Hardware systems
        - to data and applications:
            - Web
            - Mobile
            - Database

TEACHING MATERIALS

1. *"Network Security Essentials Applications and Standards", 6 th Edition, William Stallings, Published by Pearson (July 13 th, 2021) – ISBN-13: 9780134527338, Copyright © 2017*
2. *"Computer Security: Principles and Practice", 4thEdition, William Stallings and Lawrie Brown, ISBN-13: 9780134794105, 2018, ©Pearson*
3. *"Hacking Exposed", 7th Edition by Stuart McClure, Joel Scambray and George Kurtz Mc Graw Hill ISBN-10: 0071780289, ISBN-13: 978-0071780285*
4. *Official references such as Requests For Comments (RFC), available at: https://www.ietf.org*
5. *Transparencies and further in-depth material made available by the teacher on the official website of the course.*

TEACHING METHODS

The teacher will use a practical and participatory approach for the duration of the course. The topics of the course will be presented in theory and further analyzed through practical examples and demonstrations.
A fundamental tool for the course will be represented by the Docker Security Playground (DSP – https://github.com/giper45/DockerSecurityPlayground), an open source architecture based on the use of the microservices paradigm and used for the creation of virtual infrastructures specifically specialized for the study of network security. A rich set of pre-configured security labs is made available by the teacher in the form of a github repository, reachable at the URL https://github.com/NS-unina/DSP_Repo.
All students will have to work, individually or in groups, on a practical project focused on the in-depth study of one or more of the topics of the course.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

b) Evaluation methods:

- 35% of the final grade will depend on the evaluation of the practical project developed by the student (individually or in groups). This project must be delivered to the teacher (complete with documentation and including, if any, the source code developed) at least seven days before the date set for the oral exam;
- 65% of the final grade will depend on the outcome of the oral exam.

COURSE DETAILS

"CYBER-SECURITY AND DATA ANALYSIS"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ANTONIO PESCAPE'
PHONE: 0817683856
EMAIL: ANTONIO.PESCAPE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Notions of Internet network protocols.

LEARNING GOALS

The aim of the course is to provide students with specialized notions useful for the analysis of a modern internet with particular reference to aspects related to network security. The course presents the contents adopting an engineering and empirical approach and blends theoretical lessons, practical lessons, seminars and exercises. It presents in depth the main aspects and motivations behind the analysis of a computer network, and then deepens the methodological and practical aspects related to network analysis with a specific focus on the analysis, identification and classification of anomalous events such as, for example, cyber attacks. The objective is to study the main techniques, technologies and tools for monitoring and analyzing network traffic and their applications to cybersecurity. The course examines the main approaches to making Internet network measurements (passive and active monitoring), the techniques used to identify application and service traffic (traffic classification) and the methods used for network security, such as attack detection and mitigation, using machine learning approaches. The course also includes experimental activities aimed at drafting a technical-scientific report.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems related to the analysis and monitoring of Internet networks and network traffic, both benign and malicious. Must demonstrate that they can develop arguments concerning the relationships between network traffic and phenomena such as attacks, network malfunctions, performance problems. These tools will allow students to understand the causal connections between the use of network applications, the traffic generated by such applications and the operating conditions of the network in the presence of both benign and malicious traffic. In addition, the student is expected to be able to recognize the relationships between operations performed across the network (running applications, configuring network devices, attacking network devices) and observable events (changes in traffic characteristics and network device functionality).

Ability to apply knowledge and understanding

The student must demonstrate to be able to draw the consequences from a set of information to identify and solve problems concerning network infrastructures and applications, plan infrastructures and services based on the security requirements required for communications made over the Internet. By analyzing network data, the student must be able to infer the nature (benign/malicious) and volumetric characteristics of the communication (number of devices involved, quantity and timing of messages exchanged). The student must be able to apply the methodological tools learned to the following areas: collection of Internet traffic data, analysis of network applications and network security. More precisely, security analyses performed by observing and analyzing network operational scenarios constitute a proactive approach to cybersecurity, which uses data collection, aggregation and analysis capabilities in order to perform essential security functions that detect, analyze and mitigate cyber threats to networks.

PROGRAM-SYLLABUS

Introduction, Basic Concepts and Fundamentals: Didactic/Scientific Contextualization of the Course, Basic Terminology, Framing of the main aspects of Internet analysis and motivations (before and after the proliferation of cyber attacks), Network Troubleshooting, Network Requirements, Network Security, Security Analysis; Analytical background (probability, statistics, data representation, forecasting, graphs, etc.), Methodologies and Techniques of Machine / Deep Learning and Data Mining; Learning Tasks: Classification, Prediction and Anomaly Detection; eXplainable AI (XAI) techniques for Machine/Deep Learning approaches; Reinforcement Learning; Adversarial Learning; Continuous and Few Shot Learning; Evaluation Framework and Performance Metrics; Data Visualization. [1,5 CFU]

Data and Network Monitoring: Models and Metrics for Internet Analysis and Monitoring; Approaches to Internet Monitoring (active, passive, hybrid, etc.); From Network to Traffic: Methodologies and tools for the acquisition and characterization of network traffic datasets (generated by users and generated by bots); Types of data for network security (network packets, features and extracted statistics, network and system log files); Public datasets for network traffic analysis and network security: characterization and main uses; Management and configuration of network

monitoring tools; Methodologies and Techniques for Internet Traffic Analysis: network workloads, statistical characterization and modelling, Self-Similar Traffic, Internet Traffic Generation Models; "Practical Issues" in Internet Analysis and Monitoring: middleboxes (PEP, PDP, Firewall, etc). [1,5 CFU]

Data Analysis and IT Security: From traffic to data, from data to features: analysis and interpretation of network data, analysis and selection of network features; Identification and classification of network traffic with approaches based on artificial intelligence; Identification and classification of anomalous and malicious network traffic (malware, DoS/DDoS attacks, BotNets, etc.) with particular reference to 5G, IoT (Industrial), Cloud, Web and Mobile scenarios; Automatic Anomaly Identification, Zero-day Detection, Machine Learning Network Intrusion Detection Systems (ML-NIDS), new attacks on ML systems and "adversarial input perturbations", large-scale experimental platforms for the analysis and monitoring of Internet traffic and generated by attacks; Net neutrality and enforcement, detection and circumvention of censorship mechanisms; Robustness to attacks on network paths and topologies (e.g., routing protocol attacks); Legal and ethical challenges that emerge from collecting data on human user activities and using them in order to build machine learning models. [3 CFU]

TEACHING MATERIALS

The teaching material consists of slides, handouts and articles provided by the teacher.

In-depth textbook: *Internet Measurement: Infrastructure, traffic & applications, Mark Crovella, Balachander Krishnamurthy, Wiley.*

TEACHING METHODS

The teacher will use:

a) lectures for about 40% of the total hours,

b) exercises to deepen theoretical aspects for about 40% of the total hours,

c) seminars to deepen specific themes for about 20% of the total hours.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

b) Evaluation methods:

COURSE DETAILS

"SOFTWARE SECURITY"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ROBERTO NATELLA
PHONE: 0817683820
EMAIL: ROBERTO.NATELLA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

The basic notions acquired during the Bachelor's Degree in Computer Engineering on operating systems, software engineering, databases, computer architectures, and computer networks are sufficient as prerequisites.

LEARNING GOALS

The general objective of the course is to provide students with notions and skills for the design, development, validation and management of secure software systems. This objective is complementary to other courses in the "Cyber-Security" area of the Master's Degree in Computer Engineering.

The LEARNING GOALS of the course include:

- Provide advanced concepts on threats, vulnerabilities, and attacks on software systems.
- Provide specialized software design and secure development skills.
- Provide specialized skills for software validation through static and dynamic analysis.
- Provide advanced concepts on the techniques adopted in malicious software.
- Provide specialized skills for the prevention and detection of attacks due to malicious software.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of threats, vulnerabilities, and attacks in software systems, and the techniques used in malicious software. The training course aims to provide students with the knowledge and methodological tools to understand the current and future security issues of software systems in the face of their continuous evolution over time, and to effectively identify security issues and communicate them to stakeholders involved in the creation and management of secure software systems.

Ability to apply knowledge and understanding

The student must demonstrate to be able to create new secure software systems, applying the methodological and operational skills and tools acquired in the training course in the context of all phases of the software development process (analysis, design, development, testing, management). The training course is aimed at promoting the ability to make design choices of software systems in a conscious manner of security issues, and to be able to recognize and solve threats, vulnerabilities, attacks, and malicious software techniques within a software system.

PROGRAM-SYLLABUS

Introduction to software security: The secure software development cycle, vulnerability taxonomies (CVE, CVSS, CWE, OWASP), top design flaws & bugs.

Software vulnerabilities: Buffer overflows, memory & type safety vulnerabilities and other attacks (format string, double free, out-of-bounds reads), return-oriented programming, control-flow integrity, undefined behaviors, web vulnerabilities (session hijacking, CSRF, XSS, SQL/command injection).

Secure software design and certification: Software security processes and standards (Microsoft Security Development Lifecycle, Common Criteria, OWASP, CERT), principles of secure design, attack and threat modeling, security requirements and abuse cases, secure software supply chain, DevSecOps.

Secure software programming: OS/compiler/language/framework-based defense techniques, input validation strategies, regular expressions, error handling and resource management techniques.

Vulnerability identification techniques: Static software analysis (type checking, compiler-based analysis, quality scanners, security scanners), dynamic software analysis (source/binary code instrumentation, sanitizers), fuzzing (generation/mutation fuzzing, coverage-driven fuzzing, library-based fuzzing, protocol fuzzing).

Malicious software: Forms of malicious software (viruses, RAT, keyloggers, rootkits, etc.), tactics and techniques of attackers (Cyber Kill Chain, Diamond Model, MITRE ATT&CK), static and dynamic techniques of reverse engineering and analysis of binary code, malware signatures.

TEACHING MATERIALS

Textbooks:

- Wenliang Du, ***"Computer & Internet Security: A Hands-on Approach",*** 2nd edition, 2019
- A. Takanen, J. DeMott, C. Miller, A. Kettunen, ***"Fuzzing for Software Security Testing and Quality Assurance",*** 2nd edition, 2018
- B. Chess, J. West, "***Secure Programming with Static Analysis"***, 1st edition, 2007
- M. Sikorski, A. Honig, ***"Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software",*** 1st edition, 2012

TEACHING METHODS

The teaching activities will be structured in:

- Lectures, for about 60% of the total hours
- Laboratory activities to apply and deepen the acquired knowledge, for about 40% of the total hours

The lessons will make use of practical demonstrations based on virtualization environments, and on the software security tools most used in Linux and Windows operating systems.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	X
other	

b) Evaluation methods:

The evaluation takes place through the discussion of a project work, which the student develops independently alone or in collaboration with a small group of other students. The discussion aims to ascertain the acquired knowledge by the student of vulnerabilities and attacks in software systems, and his ability to operationally apply this knowledge in the design and / or validation of the security of a software system. The final evaluation takes into account the complexity of the project addressed, the quality of the software eventually developed and / or the degree of depth in the validation of the security of a software system, and the degree of maturity demonstrated in the exposure of security issues.

COURSE DETAILS

"SOFTWARE ARCHITECTURE DESIGN"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ANNA RITA FASOLINO
PHONE: 0817683906
EMAIL: ANNARITA.FASOLINO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) :
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Ingegneria del Software

LEARNING GOALS

The course aims to introduce the theme of software architectures and to provide students with specialized knowledge and skills related to the design, documentation and development of these architectures.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the problems related to the discipline of software architectures and their role in modern software development processes; The training course aims to provide students with the necessary knowledge to understand the characteristics of different architectural models, styles and patterns and the methodological tools necessary to design, document, and develop a software architecture. These knowledge and tools will allow students to understand the main relationships between the architectural project and the consequent quality characteristics of the software and to grasp the implications of the different design choices on the requirements possessed by the resulting architecture.

Ability to apply knowledge and understanding

The student must demonstrate the ability to use the knowledge and methodological tools learned to be able to design, document and implement software architectures capable of achieving all the functional and quality requirements required, implementing a development process inspired by modern iterative, evolutionary and agile processes.

PROGRAM-SYLLABUS

Software architectures: fundamental concepts.

Software Architecture Definitions. Architecture as a set of Structures and Views: Modular Structures, Component Structures and Connectors, Allocation Structures. Role and importance of architecture in software development processes.

Quality of Software Architectures.

Quality attributes of an architecture observable at run-time (performance, security, availability, functionality, usability), attributes not observable at run-time (modifiability, portability, reusability, integrability , and testability), attributes of intrinsic quality architecture (conceptual integrity, correctness, and completeness, buildability).
Specify quality attributes through the Quality Scenarios technique. (OUTLINE)

Architectural Design.

Principles of architectural design, Design based on the reuse of experience, through Patterns, Styles and Architectural Tactics.

Architectural patterns.

The Architectural Patterns State-logic-Display, Sense-Compute-Control and Model-View-Controller (MVC). Examples.

Architectural Styles

Procedural and object-oriented style, Level Style (Virtual Machines and Client-Server), Data-Flow (Batch and Pipe-and-Filter), Shared Memory (Blackboard and Rule Based), Interpreter-based (Mobile Code), Implicit Invocation (Publish-Subscribe and Event-Based), Distributed Object Style, Service Oriented Style (SOA), Microservices Style.

Tactics to realize the quality attributes of an architecture

Tactics for Availability, Deployability, Modifiability, Energy Efficiency, Performance, Security, Safety, Testability, Usability.

Architectural Solutions
Access to the resources of an architecture through the Interfaces; design of interfaces and interaction styles; the representation and structure of the exchanged resources (XML, JSON and Binary); Virtualization and Cloud Computing; Mobile applications

Document Software Architectures.
Document the different views of a software architecture: documentation styles for the modular view, component views, and connectors, for the deployment view. Document the context, behavior, design decisions, and interfaces of the components of an architecture. Overview of ISO/IEC 42010 (Systems and Software Engineering- Architecture Description).

Development of Software architectures with Iterative and Incremental processes.
Iterative and Incremental development: differences from waterfall development. Unified Process (UP): the Phases of Conception, Processing, Construction and Transition. The disciplines and practices of UP.
Agile development. The Principles of Agile Development. Agile Manifesto and Agile practices. Test Driven Development (TDD) and Refactoring. Continuous Integration (CI). The SCRUM framework: Scrum roles, Backlogs, Scrum practices.

Iterative Development and Software Configuration Management Processes
Software version management. Centralized and Distributed Version Control model. The Build process and Agile Building, Continuous Integration, DevOps. The processes for managing Software Change Requests. Release Management.

TEACHING MATERIALS

- Len Bass, Paul Clements, Rick Kazman, Software Architecture in Practice, Fourth Edition, Addison Wesley, 2022
- P. Clements, F. Bachmann, Bass, etc. Documenting Software Architectures- Views and Beyond, Addison Wesley, Second Ed., 2010.
- N. Taylor, N. Medvidovic, E. Dashofy. Software Architecture -foundations, theory and practice, Wiley 2010.

Copy of the slides projected during the lessons
Scientific articles and supplementary material from the Web

TEACHING METHODS

The teaching will be delivered mainly through lectures and will be supplemented by practical exercises and in-depth seminars. The exercises will focus on the study of real software architectures and the application of the design and documentation methods of the architectures presented in class. The performance of the exercises may require the use of integrated environments and specialized software for the design, documentation and implementation of software architectures.

The teacher will use:
a) lectures for 30 hours (equal to about 60% of the total hours),
b) exercises to deepen theoretical aspects and apply the knowledge learned for 14 hours (equal to about 30% of the total hours),
c) specialized seminars for 4 hours

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
Oral only	X
Discussion of project paper	
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

The final oral exam aims to ascertain the knowledge learned in the field of software architectures, as well as modern methods and processes of design and development of software architectures. The oral exam can be inspired by an in-depth paper carried out by students on one of the topics covered in class.

b) Evaluation methods:

COURSE DETAILS

"SOFTWARE TESTING"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: PORFIRIO TRAMONTANA
PHONE: 0817683901
EMAIL: PORFIRIO.TRAMONTANA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

The fundamental prerequisites of this course are the basic knowledge of object-oriented programming, and in particular of the Java language. In addition, a basic knowledge of Software Engineering issues is required. No prior knowledge of testing is necessary.

LEARNING GOALS

The course aims to define and deepen issues of Software Verification and Validation, including methodologies, strategies, techniques, tools and processes of Software Testing and Debugging. The course aims to contribute to the acquisition of skills related to the development of quality software and its evaluation, with particular attention to the automation of these activities.

These objectives are of fundamental importance in the realization and evaluation of any computer system and processing system, from hardware to software, from operating systems to computer networks, from databases to information systems, from artificial intelligence to robotics.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must be able to understand the importance of software quality and its evaluation, in particular the research and correction of its defects through the application of appropriate testing strategies and techniques. The student must understand what are the intrinsic difficulties in testing a software system and know how to choose between the solutions that can be applied. This knowledge will allow students to integrate knowledge related to software engineering in order to organize quality and high reliability software development processes. The student must also be able to understand how the methodologies and techniques learned can contribute to solving problems related to the realization of quality hardware or software systems.

Ability to apply knowledge and understanding

The student must demonstrate that he knows how to organize software development processes that provide for a continuous evaluation and improvement of quality thanks to the execution of effective and efficient testing activities. The student must be able to design and develop software test plans at different levels of detail, from unit testing to system testing and acceptance testing. The student must be able to carry out test cases with a high level of automation, both in the generation phase of test cases, and in the execution and evaluation phase of the outcome. The student must be able to make the best use of existing testing tools and have acquired the ability to design and implement innovative testing tools.

PROGRAM-SYLLABUS

Theoretical elements of software testing: Definitions – Undecidable problems – Taxonomy of testing activities

Quality of testing: Adequacy - Precision - Repeatability - Ability to find defects - Effectiveness - Efficiency

Specification of test cases: Input – Output – Oracles – Pre and post-conditions

JUnit: Introduction to JUnit - Implementing Unit Testing with JUnit on Java Programs – Assumptions and Assertions – Exception Testing – Dynamic and Parametric Testing – Data Driven Testing with JUnit

Black Box testing. Requirements testing and use case scenarios - Testing with equivalence classes and limit values - Tools and techniques for combinatorial generation of test cases - Testing with decision tables.

Testing White Box – Models and coverage metrics - Tools for automatic measurement of code coverage.

Integration testing and isolation testing – Isolation testing techniques with drivers and stubs – Dependency graphs – Strategies for integration testing: top-down, bottom-up, sandwich – Testing with Mock Objects – Outline of dependency injection – Framework for creating Mock Objects

User interface testing – Character user interface testing techniques - GUI testing - GUI modeling with finite state machines - State explosion problem and equivalent state technique - Libraries to support GUI testing - Input validation.

User Session Techniques: Taxonomy of Capture & Replay techniques for generating UI tests - Issues related to the generation of robust locators - Issues related to the replicability of captured tests - Capture & Replay tools for Web applications

Testing automation techniques: Automation in the generation/execution/evaluation of the outcome of test cases - Automatic generation and evaluation of oracles - Crash testing - Smoke Testing - Regression Testing.

Random testing: Features and parameters of random testing - Random testing termination problem - Techniques and tools for automatic GUI exploration - Reduction and prioritization of test suites.

Mutation Testing: Test case mutation – Mutation Analysis – Mutation based Testing: Using EvoSuite.

Experience based Testing: Exploratory Testing – Error Guessing and Checklist based Testing – Beta Testing - Crowdtesting – CrowdTesting Platforms - Software Testing Gamification – Unit Testing with Code Defenders

Testing in Continuous Integration: Notes on techniques, languages and tools for build automation - Notes on techniques and tools for managing concurrent versions - Automate testing activities in Github with Github Actions

Static Source Code Analysis: Techniques for Static Analysis - Automatic Static Analysis Tools: Checkstyle, PMD, Findbugs, Android Lint - Nod to use SonarQube for continuous monitoring of software quality

Debugging: Localization and correction of defects - Techniques for debugging: brute force - backtracking, elimination of causes - Debugging tools: breakpoints, conditional breakpoints, watches, watchpoints.

Testing of Android applications. Introduction to Android system and Android app programming - Unit testing: using JUnit and Robolectric - GUI testing: using Robotium and Android Espresso - Using Espresso Recorder for Capture & Replay of test cases - System testing: using UIAutomator - Low-level testing tools: Monkey - Monitoring tools - Memory leaks testing - Systematic testing tools: Android Ripper - Application testing with remote resources: Cloud testing with Firebase TestLab, Alpha Testing, Beta Testing.

TEACHING MATERIALS

The teaching material includes a complete set of presentations made by the teacher that accompany all the lessons of the course. These presentations contain references to numerous books, scientific articles, development tools, experimental materials, examples and other online resources, all of which are made available to students.

Recommended books include:

- Ian Sommerville, Software Engineering, Pearson Addison Wesley
- Mauro Pezzè and Michal Young, Software Testing and Analysis, John Wiley & Sons, 2008

TEACHING METHODS

The course consists of a total of 48 hours, in which frontal teaching (36 hours) alternates with presentation and discussion of the course topics and exercises (12 hours), in which techniques and software tools to support testing activities are

shown and interactively tested. For some topics, experiments are proposed based on the realization of test cases for software systems with the application of innovative testing techniques.
Recordings of all lectures (taken live while they are held), the code of all samples and exercises carried out are provided. For practical activities, only open and free software is used and the basic indications on their retrieval and use are provided.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
Other	

The exam is divided into one or two project papers, followed by an oral test. One of the two project documents consists in the design and implementation of a test plan on a specific software, possibly proposed by the teacher, applying the techniques and tools presented in the course. The second elaboration, instead, consists in the deepening of an innovative testing theme with respect to the contents of the course or in the creation of an innovative software testing tool.
The delivery and discussion with positive outcome of the project documents allows access to the oral exam.
The oral exam consists of an overall evaluation of the achievement of the expected learning objectives, through a question on topics related to the entire course program.
The final grade will take into account both the correctness and innovativeness of the project documents, and the evaluation of the oral exam.

COURSE DETAILS

"COGNITIVE COMPUTING SYSTEMS"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: PAOLO MARESCA
PHONE: 0817683168
EMAIL: PAOLO.MARESCA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide the in-depth knowledge and skills necessary for the understanding of systems based on the cognitive computing paradigm. Cognitive computing is an emerging discipline that, by bringing together knowledge of cloud, Big Data, IOT, connections between networks, machine learning, natural language processing, AI, deep learning and knowledge representation, develops automatic systems that seek to simulate the process of human thought. Students will also have the opportunity to develop the specialized skills necessary for the development of cognitive applications that can interact with people and / or things (machines and / or other computers). The course will be accompanied by an activity of exercise and development of applications in the laboratory.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must be able to know the themes and problems included in this course to the point of elaborating transversal connections between the disciplines that cognitive computing includes. The paradigm of cognitive computing is disruptive because it introduces an upheaval in the way of proceeding in solving problems and designing solutions to problems that span many themes and application domains. The course will provide the methodological tools to elaborate and apply this new paradigm on concrete cases.

Ability to apply knowledge and understanding

The student will be able, alone or in a group, to conceive, design and implement a cognitive application. He will acquire the methodological tools of the cognitive computing paradigm and apply them to the designated application domain. His ability to master the pervasiveness of cognitive applications will be important to concretely apply the knowledge gained in the course. In order to correctly implement a solution, experiences from the most quoted companies in the world will trace the maturation of cutting-edge methodological tools in companies and research centers.

PROGRAM-SYLLABUS

The program will consist of the following topics

- An overview of Artificial Intelligence and Machine Learning
- Natural Language Understanding
- Approaches to automated question answering and relation extraction
- The foundation of cognitive computing
- Design principles of cognitive computing and Design Thinking
- Cognitive computing ecosystem scenario
- Applying Advanced Analytics to cognitive computing
- Unstructured Information Management Architecture & Massively Parallel Probabilistic Evidence-Based architecture
- The Cognitive computing paradigm
- Cooperative cognitive programming & Flow-Based programming
- The role of cloud and distribute computing in CC: PaaS platform as IBM-Watson, Microsoft-Azure, et al
- Building a Watson-enabled system
- The process of Building a cognitive application
- Applications of cognitive computing to Internet of Things
- The operation of AI platforms
- Efficient use of resources
- Responsible AI
- Approach that combines small data & wide data
- New trends in Cognitive computing

An overview of artificial intelligence and machine learning

This chapter provides an overview of artificial intelligence and machine learning by tracing its track in research over the past 50 years. The current vision of AI is that of Augmented Intelligence. In other words, AI should not attempt to replace humans, but rather assist and amplify their intelligence. We can use augmented intelligence to extend human capabilities and accomplish things that neither humans nor machines could do alone. Some of the challenges we face today come from an excess of information. The Internet has led to faster communication and access to a wealth of information. Distributed computing and IoT have led to the generation of massive amounts of data, and social networking has meant that most of that data is unstructured. There is so much data that human experts can't keep up with all the changes and advances in their fields of study: we don't have a plan to use it! With augmented or expanded intelligence, we want to put the information experts need at their fingertips and return that information with a probability of correctness, so that experts can make more informed decisions. We want experts to increase their skills so that they can improve service to their customers. We want to let machines do the time-consuming work so that experts are able to do the things that matter. One of the important challenges is that artificial intelligence, replacing humans in everyday repetitive tasks, will have to be accepted in many areas of people's lives and work. The data that is collected covers people, appliances, cars, networks, healthcare, purchasing decisions, weather, communications and everything we see. It's also easier than ever to access this data thanks to smart phones and smart watches, users using it to post pictures of their groceries, store purchases and routine exercises on social media. When we develop AI systems, we need to establish trust. A problem immediately arises which is: if you do not trust a machine that provides recommendations you can not trust the advice it gives you. If someone gives you advice, we need to have a level of trust built before following it. One way to build trust is to show how the recommendation was constructed (transparency). Any solution using AI should be built with transparency and disclosure in mind. Privacy is another concern that needs to be addressed from the start when building an AI solution. Once you lose privacy, you can't get it back. Then there are the ethical problems that the use of AI creates Ethical challenges have always been there when technology has made progress: think of the early years of television etc., we are not surprised that this happens today with AI. We simply have to accept these challenges. In short, a stimulating and avant-garde world awaits us in this course.

Natural language Understanding

Natural language is the way in which human beings express concepts, needs, feelings, etc. It is the main communication tool. It seems natural and obvious to start with this powerful tool to begin a journey through study and use it to instruct machines to do what we wish them to do. But natural or spoken language has many nuances: dialects, tones, abbreviations, metaphors, idioms, etc. The natives of the spoken language have no problem speaking and writing all this but for the machines it is another story. NLP is one of the most important subfields of machine learning for a number of reasons. Natural language is the most natural interface between a user and a machine. A rigorous treatment would take away an entire course, it will be enough for us to understand what are the interesting points of the research and how language is important in designing cognitive systems that interact with the user and are able to solve problems of increasing complexity. We will see how this complexity will impact the hardware and software architectures that support applications.

Approches to automated question answering and relation extraction

In 2007, IBM Research faced the great challenge of building a computer system that could compete with the champions at the game of Jeopardy!, a television quiz show on the United States. In 2011, the open domain question answering system, called Watson, beat the two highest scoring players in a two-game match Jeopardy!

Advances in open-ended question response (QA) technology can help professionals make critical and timely decisions in areas such as: standards compliance, healthcare, business integrity, business intelligence, knowledge discovery, enterprise knowledge management, security and customer service, etc.

To carry out a project of such complexity, an adequate architecture is needed. the Qa answering system (QA) architecture designed to allow Watson-IBM to play Jeopardy! like DeepQA. DeepQA is a software architecture for in-depth content analysis and evidence-based reasoning. It represents a powerful capability that uses advanced natural language processing (NLP), information retrieval, reasoning, and machine learning. The philosophy behind the research approach that led to DeepQA is that true intelligence will emerge from the development and integration of many different algorithms, each looking at data from different perspectives. The success of Watson's question-answering system can be attributed to the integration of a variety of artificial intelligence technologies.

The DeepQA architecture sees the problem of automatic answer to questions as a task of generating and evaluating massively parallel hypotheses. DeepQA can be seen as a system that generates a wide range of possibilities and, for each, develops a level of trust by collecting, analyzing and evaluating evidence based on available data.

The primary computational principle supported by the DeepQA architecture can be summarized in the following points:

1. Assume and pursue multiple interpretations of the question.
2. Generate many plausible answers or hypotheses.

3. Collect and evaluate many avenues of competing evidence that could support or disprove such hypotheses.

The foundation of cognitive computing computing

The chapter deals with the paradigm of cognitive computing and why it is different from the deterministic one from which all the programming languages that the student knows descend. A paradigm based on data rather than algorithms and that feeds on an ever-increasing and changing amount of data that requires parallel and increasingly performing computing systems. Data changes fast. But why is this data growing so fast? because OUR SOCIAL MODEL HAS CHANGED. The keywords that describe well the model of society in which we live are: mobile, sociotechnical, complex and hyperconnected.

A cognitive technology that is based on the way of reasoning of the human being, in particular when a living being observes phenomena and makes decisions follows an approach consisting of 4 steps. For example, when we find ourselves observing something and we are called to make a decision (e.g. whether to climb the stairs or take the elevator), the mental process we follow is the following.

1) we observe visible phenomena and collect objective evidence,
2) We use what we know to interpret what we observe by generating hypotheses about what is observed
3) we evaluate which hypotheses are right and which are wrong,
4) We make the decision by choosing the hypothesis we believe to be right and acting accordingly

A cognitive system behaves in the same way and will constitute a technological and cultural revolution that will upset different sectors as it will be able to operate with big data, using the cloud, IOT and networks.

The areas of experimentation of a cognitive system are healthcare, travel transportation, economics, retail. But a cognitive system can also be used in teaching, for example to personalize learning and modify the educational path of students.

Design principles of cognitive computing and design thinking

The design of a Cognitive System requires many steps. It requires understanding the data, identifying the type of questions to be asked, and creating a corpus complete enough to support the generation of hypotheses about the domain under consideration of observed facts. In short, a cognitive system is very different from a software system because it is designed to create hypotheses from data, analyze alternatives to the hypotheses themselves and determine the availability of supporting evidence to solve a problem. It is necessary to build a new process model for cognitive applications so that it must access, manage and analyze data in a given application context, generate and measure many hypotheses (solutions) release for each supporting evidence and clues on the level of confidence, the system itself continuously updates as soon as the data is updated in order to become more and more intelligent. To design such a complex system with a strong customization on the application domain, a more agile and simple design tool is needed: design thinking. In short, design thinking, in one of its four forms, is a problem solving approach aimed at improving people's experiences to understand the needs of the user by imagining as many solutions as possible to meet his needs. The methodology is enormously creative and naturally meets with cognitive computing. Design Thinking is pushing more and more companies to change their way of innovating by designing cognitive applications in a creative way and dually the question begins to be asked: How can artificial intelligence algorithms impact Design Thinking activities?. A mini design thinking course will be delivered in this course in order to be able to design a cognitive application in a group and appreciate this innovative methodology.

Cognitive computing ecosystem scenario

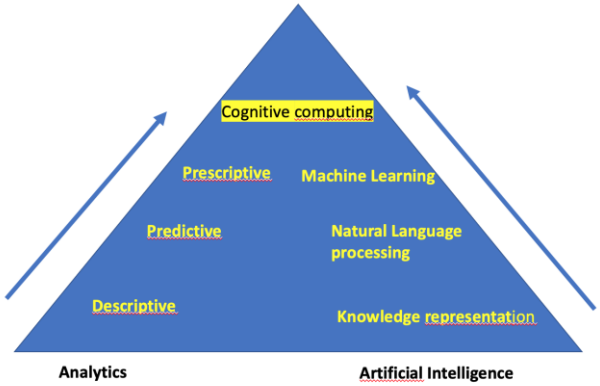
It outlines the vast ecosystem of cognitive computing. Every disruptive discipline has one

Applying advanced analytics to cognitive computing

The chapter deals with an interesting convergence between "old and new". A study conducted by the IBM Institute for Business Value entitled "Facing the storm: Navigating the global skill crisis, conducted with oxford economics questioned 5600 CEOs representing 18 industries, 48 countries and 800 leaders of government institutions as well as 1500 researchers in universities and research centers. 74% of respondents believe that companies' business models are no longer sustainable in a current market! I recommend this document, the reason? Companies will always find themselves chasing the market instead of anticipating it by observing customer needs. The maturity pattern of their processes is fragile.

As shown in the next figure, companies are experiencing a progression in the analytics maturity model, ranging from descriptive analytics to predictive analytics to machine learning and cognitive computing. The companies that have been successful are those that have figured out how to analyze data to understand both where it has been placed in the past but also how they can learn from the past to anticipate the future. They can describe how actions and events will impact results. Although knowledge of this analysis can be used to make predictions, typically these predictions are made through a lens of preconceived expectations. Data scientists and business analysts have been forced to make predictions based

on analytical models based on historical data. However, there are always unknown factors that can have a significant impact on future results. Companies need a way to build a predictive model that can react and change when the environment changes. The next frontier, a harbinger of opportunities and big changes, includes big data analytics and includes machine learning and cognitive computing technologies. As shown in the next figure there is a convergence of technologies between analysis (also conventional) and artificial intelligence. An important driver for this convergence is the change in timing and immediacy of data. Today's applications often require planning and operational changes at a rapid pace for businesses to remain competitive. Waiting 24 hours or more for the results of a predictive model is no longer acceptable. For example, a customer relationship management application may require an iterative analytics process that incorporates current information from customer interactions and provides results to support decision making in seconds, ensuring that the customer is satisfied. Therefore, analytical models must incorporate large sets that include structured, unstructured, and streamed data to improve predictive capabilities. The multitude of data sources that companies must evaluate to improve model accuracy include operational databases, social media, customer relationship systems, web logs, sensors, and video.



Unstructured Information Management architecture & Massively parallel probabilistic evidence-based architecture
In this chapter we talk about the architecture that was used by the DeepQA team to build the system that beat the best human player in the Jeopardy! tournament. So it is, in my opinion, the most brilliant invention of QA literature in recent decades.
So, the reason why IBM Watson works and why other similar systems work similarly is because the DeepQA architecture has been worked out in a very flexible way and allows the integration of a variety of different technologies including machine learning, natural language processing and reasoning and knowledge representation, and almost everything you can think of in the field of artificial intelligence.
And first of all, perhaps the most important innovation in the DeepQA architecture is the fact that we can simultaneously use both structured and unstructured data in the same architecture. The architecture is cropped, this is also an important innovation, depending on the domain for example we can have a simplified version of this DeepQA architecture which is what we call the minimum DeepQA pipeline. Architecture itself is typically parallel because it has to explore all avenues (in parallel) as it has an evidence-based probabilistic approach.

Cooperative computing programming & Flow-based programming
Cooperation and collaboration are the fundamental challenge for a working group that is also very large and geographically decentralized. Interesting programming tools to activate cooperation, distribute tasks, control work and coordinate activities, so that the effort converges in the direction of amplifying efforts in a collaboration of the development team. Although the designer of cognitive systems writes little code and has to reuse parts of it and coordinate many libraries this makes it essential to use mature platforms for development. Eclipse is one of these for its ability to host all AI languages (phyton, R, etc) as well as all the more traditional programming languages with which many algorithms are written (c, c++, c #, java, javascript, ruby, etc). Up to the languages typically used in AI such as python that will be used in this course [4]. It has the advantage of being able to intervene with a single IDE for any language and maintain a standard of cooperation during development. Flow-based programming is the easiest and easiest way to combine the advantages of a workflow and a dataflow when designing an AI application. In this course we will use Node-RED which is a flow-based programming tool[3].

The role of cloud and distribute computing in CC: PaaS platform as IBM-Watson, Microsoft-Azure, et al

The ability to leverage highly distributed and affordable computing services has not only transformed the way software is managed and distributed today, but has also become a mainstay of cognitive computing. Massive cognitive computing systems require a converged computing environment that supports a variety of types of hardware, software services, and network elements that must be workload-balanced. Therefore, cloud computing and a distributed architecture are the basic models needed to operationalize cognitive systems at scale.
We review the major Platform as a Service such as IBM Watson, Microsoft Azure, Google Ai Platform, etc. Well knowing that the scenario, at the moment, shows at least twenty cognitive computing platforms. The chapter tries to outline criteria for choice "oriented" to problems and domain. In order to make the choice objective but also by observing other criteria such as costs, data processing and their keeping and many other criteria that could be priority in some contexts.

Building a Watson-enabled system
It defines, defines a domain, an original way to cut an architecture of a cognitive system on a problem, with all that this entails in terms of balancing cloud resources, computation, memory and choosing the appropriate machine on which all this must be implemented and must be executed. This is done by taking as a reference one of the most widespread IBM-Watson cogitive systems.

The process of Building a Cognitive application
Building a cognitive application is not the same as building a software application. Cognitive systems are disruptive and upset design paradigms, hence the need to design a process model for the development of cognitive application that differs from that of software development in many parts and, even, in some is even more creative and complete. Defining goals, defining the domain and understanding the real intentions of the user and hidden clues are a very in-depth discipline in software engineering, but the discipline of cognitive systems engineering needs simpler and shared tools for the high stages of design for this we will use design thinking tools that help to share specifications and accept domain constraints. The definition of the questions and the exploration of the clues together with the acquisition of the data sources that will constitute the corpora will be the basic element on which the whole cognitive system will revolve. The corpora, and this is an important difference, is always alive: once created it must be continuously updated because it is on this that the application is updated. Finally, the training and testing techniques of the system are to complete this chapter.

Application of cognitive computing to internet of things
Cognitive applications meet the internet of things and the management of heterogeneous data in real time, building specific applications for solving increasingly complex problems. The management of a smart city, the management of a cognitive application to improve the health of a patient and his well-being are two examples of applications that pass through the collection of data that come from sensors and that must be processed in time to allow "decision-makers" to activate the right decisions to control problems.

The operation of AI platform
For most organizations, the integration of artificial intelligence solutions within business workflows represents, to date, a complex job, studded with failures. Only half of AI projects move from pilot to production. And, in the latter case, on average it takes about eight to nine months to get to the complete development of an efficient and integrated AI system within its operations, giving this system a tangible value in business transformation. Well, Gartner predicts that, by 2025 – thanks to the increasing maturity of AI Orchestration and Automation Platform technologies – 70% of companies will have made their artificial intelligence architectures operational, thus moving AI projects from idea to production and making them, in practice, useful.

Efficient Use of resources
Investing in artificial intelligence means making efficient use of all available resources, including data and calculation models. An example comes from composite AI that combines different techniques, including deep learning, graph analysis, agent-based modeling and optimization techniques. With the result of a "composite" artificial intelligence system, able to solve a wider range of business problems. But it is necessary that these technologies (and the data used to train the algorithms) are used with maximum efficiency by those who possess the skills.

Responsible AI
"Greater transparency and auditability of AI technologies continues to be crucial. This is responsible AI." All the more necessary the more AI comes to replace large-scale human decisions, amplifying the positive and negative impacts of these decisions. The focus must be on the data that is fed to the machine. If the data is "good", free from prejudices, it will train an equally "good" and bias-free algorithm. Attention, in particular, must be high towards "implicit" prejudices, less obvious and, therefore, more difficult to identify. Such as, for example, those that lead to decisions that discriminate

against age or gender. In the coming years, therefore, organizations will need to be able to develop and manage artificial intelligence systems "that are ethical and transparent" and, to achieve this goal, all staff dedicated to the development of AI systems demonstrate experience in "responsible AI".

Approach that combines small data & wide data

In terms of new trends in artificial intelligence, another trend concerns "small data", i.e. those data that have to do with the application of analytical techniques that require less information. This data together with the use of large data sets (wide data or big data), allow more in-depth analysis and help to obtain a broader view of the problem to be solved by means of AI. The Hype Cycle for Artificial Intelligence 2021 indicates that, by 2025, 70% of organizations will be forced to shift their focus from large to small data, thus giving more space to data analysis and cross-referencing. This is a trend observed since the pandemic crisis, which has caused a rapid decline in the large amounts of historical data, related to past situations, and thus breaking previous patterns. Adopting analysis techniques that combine "small data" and "wide data" means, instead, working with different volumes of data and extracting value from different and unstructured sources.

TEACHING MATERIALS

Textbooks:

- E. Kelly III and S. Hamm, Smart Machines IBM's Watson and the Era of Cognitive Computing (2014), ISBN: 978-0-231-16856-4.
- Alfio Gliozzo et al, Building Cognitive Applications with IBM Watson Services: Volume 1 Getting Started, (2017) IBM redbook , https://ibm.co/30V63PU
- J. Hurwitz, M. Kaufman, A. Bowles, Cognitive Computing and Big Data Analytics (2015), ISBN: 978-1-118-89662-4.
- Taiji Hagino, Practical Node-RED Programming (March 2021), ISBN 978-1-80020-159-0.
- Paul Deitel, Harvey Deitel, Intro to Python for Computer Science and Data Science, (2020), ISBM-13: 978-0-13-540467-6.

Paolo Maresca placeholder image Lesson material and course recordings a.y. 2020-2021 COGNITIVE COMPUTING SYSTEMS access on Teams with code tr9xxpl link: https://bit.ly/2SBD0t3

TEACHING METHODS

The teacher will use lectures for about half (50%) of the course, the remaining part will be dedicated (50%) to exercises, seminars and laboratories. The workshops will serve to deepen the theoretical and methodological aspects that emerged in the lectures. Given the strong correlation of the course with the evolution of cognitive computing, the laboratories may be different from year to year and will also be called upon to hold the most innovative topics. Seminars on new trends in cognitive computing are also planned.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	**X**
Discussion of project paper	**X**
other	

COURSE DETAILS

"BUSINESS PROCESS AUTOMATION"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: VALERIA VITTORINI
PHONE: 0817683847
EMAIL: VALERIA.VITTORINI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : II
SEMESTER (I, II): II
CFU: 3

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

For a better use of the course contents, the student should have basic knowledge of programming and XML and Java languages.

LEARNING GOALS

The aim of the course is to provide students with the main concepts related to workflow management. The focus of the course is on the definition, representation and coding of the workflow, through the use of languages, such as Business Process Modeling Notation (BPMN), and workflow patterns. Case studies are presented on modern cloud-based systems and services (e.g., Netflix, AWS).

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate that he has acquired the basic knowledge necessary for the definition and management of workflows in different application domains.

Ability to apply knowledge and understanding

The student must demonstrate autonomy in the use of technologies and tools in the definition and execution of simple business processes, through one or more platforms introduced during the course.

PROGRAM-SYLLABUS

BPM and Workflow. Framework in the context of Business Process Management (BPM). Definition, classification of business processes, life cycle. Workflow: BPM and workflow, workflow definition, fundamental concepts, reference architecture of a workflow management system. Process model: events, triggers, routing operators. Workflow data. Organizational model, classification of resources. Languages for defining business processes.

Orchestration and Choreography. SOA, REST, microservices: fundamental principles. Automation by orchestrating services.

Automation and Application Integration: Layers. Integration patterns and patterns. Key components: Adapters, wrappers, brokers. Application integration architectures: Hub and Spoke, Bus, Middleware, Enterprise Service Bus. Workflow Patterns.

Case studies. Netflix, AWS.

TEACHING MATERIALS

Course notes, scientific articles, application documentation used as a case study.

TEACHING METHODS

Lectures (40%), exercises (30%), seminars (30%)

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	x
other	

COURSE DETAILS

"IMAGE PROCESSING FOR COMPUTER VISION"

SSD ING-INF/03

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: GIUSEPPE SCARPA
PHONE: 0817683768
EMAIL: GIUSEPPE.SCARPA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): I
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Teoria dei Segnali

LEARNING GOALS

The course aims to provide students with in-depth notions on the development and application of image processing techniques for the solution of typical *computer vision* problems, ranging from traditional methods for signal processing, i.e. modeling-oriented, to modern approaches based on convolutional neural networks. Specific computer *vision* problems considered as training objectives of the course are the detection, characterization and *matching* of local *features*, *fitting* and alignment of geometric models, image classification, semantic segmentation or for image instances, detection, localization and recognition of objects, estimation of pose, depth estimation, stereo matching, 3D reconstruction from multiple views.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student will have to know both classical filtering techniques and approaches based on modern convolutional neural networks for the solution of computational vision problems such as detection, description and *matching* of local *features*, *fitting* and alignment of geometric models, image classification, semantic or instance segmentation, detection, localization and recognition of objects, the estimation of the pose, the estimation of the depth, the stereo correspondence, the 3D reconstruction from multiple views, from the perspective of signal processing. For the problems listed, the student will also have to know the metrics or performance indices useful for evaluating possible solutions.

Ability to apply knowledge and understanding

The student must acquire the ability to design, develop and test state-of-the-art image processing algorithms aimed at solving common computational vision problems, including the detection, description and *matching* of local *features*, the *fitting* and alignment of geometric models, the classification of images, semantic or instance segmentation, object detection, localization and recognition, pose estimation, depth estimation, stereo matching, 3D reconstruction from multiple views.

PROGRAM-SYLLABUS

Reminders about image filtering. Space-scale domain and pyramidal decomposition. Reminders on programming environments for the development of *computer vision* algorithms.

Image formation: Light and color. The *pinhole camera* model. The projection of the 3D world in the image plane: camera projection matrix and camera calibration. Projective geometric transformations.

***Early vision*:** Outline detection; segmentation by *watershed* transform; *template matching* and textural description; detection of angles (Harris *detector*) and lines (Hough transform).

Keypoint detection and description: Definition of *keypoints* and repeatability properties. Invariance properties of detectors with respect to illumination, translation, rotation, scale, affine transformations and homographs. Harris detector. Gaussian difference (DoG). Pyramid of DoG. Orientation and scale of a *keypoint*. Feature descriptors: discriminative properties; commonly used descriptors (SIFT, SURF, MSER,...); form and context descriptors.

***Matching*, *fitting* and alignment:** *Matching* of *features* using distance ratio criteria. *Fitting* and alignment: linear or robust least squares method; ICP algorithm; generalized Hough transform; RANSAC algorithm. Detection, recognition and classification.

Image processing using convolutional neural networks (CNNs): Convolutional architectures for image processing. CNN training for image processing: backpropagation and SGD optimization algorithm (and variants). Commonly used layers: convolution, *pooling, unpooling*, *batch normalization*, activation functions (ReLU and its variants, Tanh, sigmoid). Cost functions for image processing. *Dropout* and data *augmentation*. CNN models for super-resolution, classification, segmentation, object detection and localization, depth estimation, laying estimation.

Multi-view vision: Stereo vision: disparity and depth. Epipolar constraints; Essential matrix and fundamental matrix. Dense matching problems. 3D reconstruction from *multi-view*: *Structure from Motion* (SfM).

TEACHING MATERIALS

- R. Szeliski, "Computer vision: algorithms and applications", Springer 2010.
- R.-I. Hartley, A. Zisserman, "Multiple View Geometry in Computer Vision", C. U. P., 2nd ed., 2004.
- I. Goodfellow, et al., "Deep Learning", MIT Press, 2017.
- Lecture notes.

TEACHING METHODS

The course includes both lectures (about 60% of the total) and laboratory activities. Among other things, there will be introductory tutorials both on the Python programming language and the attached *toolboxes* for *deep learning* and on the use of cloud computing platforms functional to the objectives of the course. Part of the laboratory hours will be dedicated to the ongoing development, with tutoring, of the students' projects for the final evaluation.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	
other	x

The exam includes the presentation of a project carried out individually or in groups, with relative discussion, and a general interview on the contents of the course. The project is normally developed in itinere and presented at the end of the course in a closing *workshop* , while the interview can be held in any appeal of the current academic year without time constraints.

COURSE DETAILS

"ECONOMIA E ORGANIZZAZIONE AZIENDALE"

SSD ING-IND/35

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: PIERLUIGI RIPPA
PHONE: 0817683934
EMAIL: PIERLUIGI.RIPPA@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Basic knowledge of Analisi Matematica I

LEARNING GOALS

The aim of the course is to provide students with knowledge related to principles, best practices, regulations and The course aims to provide the fundamental concepts and tools to understand the functioning of an economic and organizational system starting from the microeconomic perspective, and subsequently providing the strategic perspective of the organization both from an external and internal point of view.
As far as the microeconomic perspective is concerned, the main problems related to consumer and business behavior will be analyzed and deepened, with a focus on the main forms of market.
As far as the perspective of the organization is concerned, the external condition will be deepened through the learning of the tools for analyzing the external environment for the definition of the business strategy, and the principles of the business model and the business plan will be introduced.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student will have to acquire the vocabulary necessary to describe economic phenomena. He will have to demonstrate knowledge of the fundamental concepts of consumer and producer theory. The student will have to demonstrate to understand how organizations define business strategy based on the analyzes carried out on the environment in which the organization lives.

Ability to apply knowledge and understanding

Based on the acquired knowledge, the student must be able to analyze the functioning of an organization through the simulation of a startup project in which all the concepts learned will be applied.

PROGRAM-SYLLABUS

Problems and techniques associated with the analysis, design, development and verification of systems used in INTRODUCTION TO THE ENTERPRISE (Definition of enterprise, costs and business objectives, the classification criteria of enterprises)
MICROECONOMICS (The enterprise and the market, supply and demand, market forms)
STRATEGY (Definition of sector; sector analysis – PORTER model; industry life cycle; Basic competitive strategies; Company positioning – SWOT analysis)
BUSINESS PLAN (executive summary, operational plan, organizational plan, marketing plan)

TEACHING MATERIALS

- Handouts and materials made available by the teacher
- "The job of doing business. Entrepreneurs are not born, they become!". By Michele Raffa, Edizioni Scientifiche Italiane

TEACHING METHODS

Lectures, exercises, external seminars, simulation of a business project (group work)

EXAMINATION/EVALUATION CRITERIA

(b) Evaluation procedures

Individual evaluation in thirtieths on the written test (mandatory)
Group evaluation on the presentation of the project

COURSE DETAILS

"SAFETY-CRITICAL SYSTEMS"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: VALERIA VITTORINI
PHONE: 0817683847
EMAIL: VALERIA.VITTORINI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 3

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide students with knowledge related to the principles, best practices, regulations and processes for the design of safety-critical systems, with particular reference to the modeling and analysis of these systems and formal verification techniques.
The role and importance of formal methods in the development of safety-critical systems will be illustrated and several formal tools used for system and property modeling will be introduced. Finally, advanced aspects will be addressed, in particular in the field of modeling methodologies of complex systems.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the concepts underlying the development of safety-critical systems and the role of formal models for the analysis of critical systems, and in particular for property verification and validation.

Ability to apply knowledge and understanding

The student must demonstrate the ability to apply the concepts and techniques presented in the course to simple case studies, also developing and analyzing simple formal models.

PROGRAM-SYLLABUS

Part I: Problems and techniques associated with the analysis, design, development, and verification of systems used in safety-critical applications. Elements of design and development of safety-critical systems (architectures, techniques for the development of safety-critical systems, e.g.; formal testing, high-integrity programming, software inspection), European standards.
Part II: The role of formal methods in systems engineering, formal methods in the certification of real systems, some examples from the real world, functional and non-functional properties, qualitative and quantitative analysis. Petri nets and timed extensions for quantitative analysis of temporal properties, formal languages for specification and analysis. Time logic, LTL and CTL, introduction to model checking. Complex model development techniques, tools for modeling and solving models.
Part III: exercises and application to case studies.

TEACHING MATERIALS

Course notes, scientific articles.

TEACHING METHODS

Lectures (60%), exercises and laboratory activities (30%), application seminars (10%).

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	x
Other	

COURSE DETAILS

"CIRCUITI PER DSP"

SSD ING-INF/01

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: DAVIDE DE CARO
PHONE: 0817683136
EMAIL: DAVIDE.DECARO@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): I
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Basic knowledge of the functioning of digital circuits and of the C language.

LEARNING GOALS

The course aims to provide students with advanced and high-level notions related to the implementation of digital signal processing algorithms and, moreover, to acquire knowledge, both basic and advanced, related to the architectures of commercially available DSP circuits, of the problems, both theoretical and practical, related to the optimal implementation, in real time, on DSP, of the main digital signal processing algorithms and of the development environment for their programming.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

At the end of the learning process the student is informed about the main techniques of data processing in fixed and floating point, on the tools for analyzing representation errors and overflow problems in a linear system realized in fixed-point arithmetic, on the main techniques of prevention / management of overflow, on the theoretical problems related to the description in C language of digital signal processing algorithms, on the minimum architectural characteristics that identify a DSP (Digital Signal Processor) circuit, on the problems related to the pipeline in DSPs, both in the case of protected pipeline and in the case of unprotected pipelines, on the characteristics of DSPs with high processing parallelism -especially based on Very Long Instruction Word (VLIW) architecture-, to the forms of parallelism at the level of the instruction set of the DSP -instructions SingleInstructionMultiple-Data-, on code optimization techniques for VLIW architecture -in particular using Loop Unrolling and Software Pipelining techniques-, on the basic and advanced characteristics of interrupt and DMA (DirectMemoryAccess) systems in DSPs, on the characteristics of DSP communication interfaces -especially synchronous serial ones-, on the problems related to the realization of time processing systems on a DSP real -especially using streaming or block processing-.-

Ability to apply knowledge and understanding

At the end of the learning process, the student is able to design, autonomously, the implementation in fixed and floating point arithmetic, of a signal processing algorithm taking into account both the representation errors and the problems related to the overflow and to determine the overflow prevention / management approach that best suits the specific application.

The student also acquires the ability to describe in C language a digital signal processing algorithm with particular reference to the problems related to the efficient implementation on DSP both fixed point and floating point. The student also masters code optimization techniques, with particular reference to VLIW architectures and is able to manage in the best possible way, in relation to the specific application, the tradeoff between calculation time and code size by applying the techniques of Loop Unrolling and Software Pipelining.

Finally, the student masters the approaches for real-time processing of a digital processing algorithm on a DSP, and, in relation to the characteristics of the specific application, can solve in the best possible way the tradeoff between maximum processing frequency and latency by adopting streaming or block processing approaches. The student also masters the problems inherent in the use of DMA units both to support processing and to service the interfaces of the DSP in block processing.

Making judgements, Communication skills, Learning skills

In relation to all the skills described above, the student also acquires autonomous judgment skills being, among other things, in the particular context, an essential prerequisite of the creative activity of a design type that is part of the ability to apply the knowledge that are developed.

During the course students are also stimulated in the acquisition of tools that allow the autonomous deepening of the topics covered, while the methodologies of verification of learning by students tend to develop their communication skills.

PROGRAM-SYLLABUS

Introduction: Digital Processing Circuits: ASIC, FPGA, DSP. Real-Time processing.

Numerical problems in signal representation: References on Fixed-Point and Floating-Point representations; Quantization Accuracy and Errors; System-level quantization errors; Overflow and prevention techniques: Representation Variation, Scaling, Scaling for Lowband signals, Statistical Scaling, Saturated Arithmetic, Guard Bits; Application of Overflow prevention techniques to a FIR filter. Description of DSP algorithms in C language: Generalized Fixed-Point representation, Type Promotion Rules, Example (FIR filter with Gaurdia bit technique).

Basic architectures of DSP circuits: Datapath; Modified Harvard and Harvard architecture; Memory in DSPs: Separate Banks, Interveaving Banks, Dual-ported Memories, Data Alignment and Multiple Access, Memory Hierarchies, Caching Consistency and Predictability; Repeat Buffer; Addressing for DSPs and Address Generation Units: Circular and Bit-Reversal addressing; Zero-Overhead-Looping techniques; Simple peripherals in DSPs: Timers/PWM generators; Texas Instruments DSP examples: C2xx, C54xx, and C55xx architectures.

Advanced DSP Architectures: Trade-off between Orthoganality and Code Dimension; Pipelining in DSPs: Unprotected or Visible Pipeline, Protected or Transparent Pipeline, Structural Hazards, Data and Dependency Hazards, Control Hazard, Delayed-Branch; Static and Dynamic Scheduling of Instructions; Superscalar architectures (outline); Very-Long-Instruction-Word (VLIW) architectures; Non-Secure Pipeline and Interruptions (trade-off between code breakability and compute time); Instructions and Single-Instruction-Multiple-Data (SIMD) Arithmetic; Texas Instruments VLIW DSP: C64xx, C67xx, C66xx architectures. Overview of the main Analog Devices DSPs (Blackfin, SigmaDSP, SHARC and TigerSHARC) and Freescale (Symphony, StarCore SC3400 and SC3850). DSP with Multi-threading Hardware: Qualcomm's Hexagon architecture. Texas Instruments' DSP-based SoCs: DaVinci Digital Media Processor, OMAP, Keystone and Keystone II.

Texas Instruments C64xx and C67xx DSP Architecture: VelociTI Architecture; Instruction set; Memory Architecture and Caching on two levels; conditional execution; Offset addressing; Circular Addresses and AMR Register; DSP Pipeline: Delay-Slots and Instruction Latency; SIMD instructions.

Code Development and Optimization for DSP VLIW: Assembly Development: Dependency Graphs, Instruction Parallelization, NOP Elimination, Loop Unrolling; Software Pipelining: Minimum-safe-trip-count, Resource-Bound, Loop-Carried-Rosource-Bound, Speculative Execution, Live-too-Long Issues, Joint Use of Loop Unrolling and Software Pipelining; Code development in Linear-Assembly; C Code Development: Software Pipelining and Unrollimg Directives, Pointer Aliasing, Memory Allocation Directives, Optimization Levels, Static Functions, Code Interruptability, Software Pipelined Loop Buffer (SPLOOP); Linker, Global and Static Variables, Near and Far variable allocation, DSP Start-up.

Interrupts and DMA in Texas Instruments C6x DSPs: Interrupts: basic functionality, interrupt handling registers, detailed operation in Hardware, writing the Interrupt-Service-Procedure (ISR), characteristic timing; DMA: basic functionality, parameters and types of transfer, synchronization, Quick-DMA, generation of completion interrupts, examples.

Real-Time implementation of DSP algorithms: Synchronous Serial Interfaces and McBSP interface of Texas C6x DSPs; Management of communication in Polling, Interrupt, DMA; Interruptible and Non-Interruptible Streaming Processing, Flow Control in Receive/Transmit processing; Real-Time Operating Systems and Pre-emption Processing; block processing: ping-pong buffering; Buffering Techniques for Memory Algorithms; Off-line and real-time debugging in DSPs, In-System Debugging techniques via JTAG interface.

Tutorials: Tutorials leverage Texas Instruments' Code Composer Studio development environment and DSK-C6713 rapid prototyping boards.
FIR Filter Implementations: Coefficient Calculation, Numerical Design and Overflow Issues, Code Optimization with Unrolling examples, Software Pipelining, Data Buffering; Experimental tests with Real-time I/O from the audio codec on the DSK-C6713 board: FIR filter, 5-band audio equalizer, 5-band audio equalizer using real-time Operating System, 9-band audio equalizer using real-time Operating System.

TEACHING MATERIALS

- John G. Proakis, Dimitris G. Manolakis, "Digital Signal Processing: Principles, Algorithms and Applications", 4th edition, Prentice Hall 2007
- Sen M. Kuo, Woon-Seng Gan, "Digital Signal Processors: Architectures, Implementations, and Applications", Prentice Hall 2005
- Lecture notes
- Tutorial texts

TEACHING METHODS

The course includes lectures, exercises and, compatibly with the organizational aspects, laboratory lessons.
Students use Texas Instruments' Code Composer Studio development environment to conduct the exercises. The laboratory, in addition to Code Composer Studio, involves the use of boards for rapid prototyping (DSK6713 by Texas Instruments) and the basic instrumentation of an electronics laboratory (oscilloscope, signal generator).

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	✓
Discussion of project paper	
Other (Tutorial discussion)	✓

COURSE DETAILS

"MODELLI E ALGORITMI DI OTTIMIZZAZIONE"

SSD MAT/09

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: CLAUDIO STERLE
PHONE: 0817685911
EMAIL: CLAUDIO.STERLE@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 9

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to provide students with advanced knowledge of mathematical programming for the modeling and exact resolution of complex network optimization decision problems in engineering. The theoretical study of the main algorithms for the calculation of the optimal solution of the decision-making problems addressed is completed by the numerical experimentation of these algorithms through the use of optimization software. At the end of the course the student will have acquired the knowledge of advanced methodologies for the modeling and solution of continuous, whole and mixed-whole optimization problems on computer and telecommunications networks.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The training course aims to provide students with the methodologies of continuous, whole and mixed-whole optimization necessary for the modeling and exact resolution of engineering problems in the field of computer and telecommunications networks. The student must demonstrate that he has acquired the tools necessary to formulate a decision problem through a mathematical programming model, identifying objective function, decision variables and constraints of the system under study. The student must also be able to identify the best solution method to be used for determining the optimal solution of a decision problem, in relation to its specific characteristics. Finally, the student must be able to analyze the sensitivity of the solution obtained with respect to the variability of the boundary conditions and understand the causal links.

Ability to apply knowledge and understanding

The training course is aimed at transmitting the methodological and operational tools necessary to concretely apply the knowledge of mathematical programming to emerging network optimization problems in the IT and telecommunications context. In particular, the student must demonstrate that he knows how to develop all the phases of a decision-making process: analysis of the system (definition of its components, of the parameters that characterize it, assumptions and operating specifications); definition of the decision-making problem; selection/construction of a mathematical model for system simulation; implementation and resolution of the model through an algorithm and/or optimization software; analysis and interpretation of the results in order to verify the quality of the solution and put in place any feedback mechanisms.

PROGRAM-SYLLABUS

- Unconstrained multidimensional nonlinear optimization
    - Gradient methods
    - Steep descent and ascent algorithm, conjugate gradient
    - Graphical analysis and numerical exercises

- Constrained multidimensional linear and linear optimization
    - Optimal conditions in constrained optimization problems (Kuhn-Tucker conditions)
    - Lagrangian relaxation
    - Methods with permissible direction
    - Graphical analysis and numerical exercises
    - Linear optimization as a special case of nonlinear optimization
    - Simplex algorithm, stability and duality analysis

- Advanced integer linear optimization (PLI) methods

- Formulation of problems integer linear optimization and convex core
- Advanced resolution methods based on "row and column generation"
- Branch and Bound and Branch and Cut
-Relaxation techniques

- Advanced problems of routing, localization and network design (modeling and solution).
- Route problems: minimum path and minimum constrained path problems (minimum paths through specified vertices, with time windows, with limited resources), minimum path with capacity constraints (Quickest path and quickest flow); Maximum path problem

- Flow problems: single and multi-commodity flow problems with constant costs and variable costs; maximum flow problems; problems with unicast and multicast streams;

- Network design issues: localization issues; design of multi-layer networks; design and dimensioning of communication networks; Designing reliable and resilient networks.

- Optimization software:
- introduction to optimization software (Xpress, Cplex);
- modeling and solving real problems of continuous, integer and mixed-integer linear programming using exact techniques

TEACHING MATERIALS

- A. Sforza, Models and Methods of Operations Research, III ed., ESI, Naples

- F. S. Hillier, G. J. Lieberman, Operations Research - Fundamentals, 9/ed., McGraw-Hill

- C. Guéret, C. Prins, M. Sevaux, Applications of optimization with Xpress-MP, Editions Eyrolles,Paris

- IBM ILOG CPLEX V12.7 User's Manual for CPLEX

- Supplementary teaching material provided during the course and material available online

TEACHING METHODS

The teacher will use: lectures (60%), seminars (10%), numerical exercises and introduction to the use of optimization software (30%). The course material will be made available online to students.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

In the case of a written test, the questions are (*)	**Multiple choice**	
	Free answer	
	Numerical exercises	

The exam involves the development of a project in which the student must develop and solve a mathematical programming model (continuous, whole or mixed-whole, representative of a real decision-making problem. The developed model must be implemented in an optimization software (Xpress or Cplex) and solved through the use of the relevant libraries of exact methods. The project paper is assigned to the student before the end of the course and must be delivered before the oral interview. The oral interview will have as its object both the discussion of the project and the assessment of the acquisition of the concepts and methodologies illustrated during the lessons.

(b) Evaluation procedures:

The delivery of the project paper is binding for access to the oral interview. The project and the oral exam each contribute to 50% of the final evaluation. The delivery of the project is not sufficient to pass the exam.

COURSE DETAILS

"RISK ASSESSMENT"

SSD ING-INF/05

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ALESSANDRA DE BENEDICTIS
PHONE:
EMAIL: ALESSANDRA.DEBENEDICTIS@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The course aims to introduce the process, the main methodologies and techniques for risk assessment in safety- and security-critical systems.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate knowledge and understanding of the issues relating to the identification, assessment and management of risk in different contexts.

Applying knowledge and understanding

The student must demonstrate to be able to apply the main risk assessment techniques seen in the course to real case studies.

COURSE CONTENT/SYLLABUS

INTRODUCTION AND TERMINOLOGY: Definitions of risk; accident scenarios; hazards and threats; hazardous events; failures and faults; likelihood and consequences; consequence spectrum; barriers and escalation factors. Representing accident scenarios with the BowTie model. Goal of risk analysis and risk assessment.

RISK MANAGEMENT PROCESS OVERVIEW: Requirements and characteristics of an effective risk management process. Overview of the main phases of the risk management process. Risk management and decision-making: deterministic decision-making, risk-based decision-making; risk-informed decision-making. Risk management in the safety and security-related regulation.

RISK ASSESSMENT PROCESS OVERVIEW: Overview of the main phases of the risk assessment process. Study object definition: systems, system boundaries; system breakdown.

RISK ACCEPTANCE CRITERIA: Risk acceptance principles: equity, utility, technology. The ALARP approach: principles and examples. Overview of other approaches: ALARA, SFAIRP, GAMAB, MEM. Cost-benefit analysis: evaluating costs and benefits: value of a statistical life (VSL) and value of prevention of a fatality (VPF).

RISK MEASURE: Risk metrics and measures; risk metrics vs performance metrics; Individual risk metrics: Average Individual Risk (AIR), Location-specific Individual Risk (LSIR), Lost-time injury (LTI) frequency, Lost workdays frequency (LWF); Group risk metrics: Potential Loss of Life (PLL); Fatal Accident Rate (FAR), FN curves; Risk matrices; Risk Priority Number (RPN).

HAZARD IDENTIFICATION METHODS: Hazard causal factors; System safety hazard analysis types: CD-HAT, PD-HAT, DD-HAT, SD-HAT, OD-HAT, HD-HAT, RD-HAT. CD-HAT techniques: checklist methods and preliminary hazard lists (PHL). PD-HAT techniques: preliminary hazard analysis (PHA); HAZID. DD-HAT techniques: subsystem hazard analysis (SSHA); HAZOP. SD-HAT techniques: system hazard analysis (SHA); Operating and support hazard analysis (O&SHA); Failure modes, effects and criticality analysis (FMECA): identification of failure modes, rates, causes, effects. Qualitative and quantitative criticality analysis. Petri network analysis: PN elements, enabling and firing rules; reachability set; properties; structural and behavioral Petri net analysis; Timed Petri nets; Stochastic Petri nets; generalized stochastic Petri nets;

CAUSAL AND FREQUENCY ANALYSIS METHODS: Fault tree analysis: FT diagrams elements; FTs and reliability block diagrams; Cut sets and minimal cut sets; MOCUS algorithm; analytical calculation; importance metrics. Cause and effect diagram; Bayesian Network Analysis: BN definition; quantitative analysis: assumptions; evidence; Common-cause failure analysis: common-cause failures, dependency, methodology steps.

DEVELOPMENT OF ACCIDENT SCENARIOS: Event tree analysis: pivotal events, methodology steps; cause-consequence analysis.

SECURITY RISK ASSESSMENT: Security risk concepts overview: main security properties, threats, vulnerabilities, weaknesses, attacks, security controls. Security risk: likelihood and impact.

THREAT MODELING: Threat modeling and threat intelligence; MITRE ATT&CK Framework. System modeling: Data Flow Diagrams (DFD) and Process Flow Diagrams (PFD). Threat identification and assessment methodologies and frameworks: The STRIDE methodology and the Microsoft Threat Modeling Process. The Microsoft Threat Modeling Tool (TMT). The DREAD methodology. The LINDDUN methodology. The Trike methodology. The VAST methodology. The CVSS framework. The OWASP Risk Rating Methodology. Attack graphs: modeling and tools.
A STANDARD-BASED SECURITY RISK MANAGEMENT APPROACH: FISMA: security categorization (FIPS-199 and NIST SP 800-60) and minimum security requirements (FIPS 200 and NIST 800-53); The NIST Risk Management Framework (NIST SP 800-37); NIST Risk Assessment Process (NIST SP 800-30); NIST Cybersecurity Framework: core, tiers, profiles.

READINGS/BIBLIOGRAPHY

Textbook: Marvin Rausand, Stein Haugen. Risk assessment - Theory, Methods and Applications. Second edition Wiley.
Lecture notes and presentations provided by the teacher relating to theoretical and applicative topics covered in the course.

TEACHING METHODS

The course includes about 70% of lectures in which theoretical topics are addressed, while the remaining 30% is reserved for exercises and seminars by experts in the development and management of critical systems.

EXAMINATION/EVALUATION CRITERIA

b) Exam type:

Exam type	
written and oral	
only written	
only oral	
project discussion	x
other	

In case of a written exam, questions refer to: (*)	**Multiple choice answers**	
	Open answers	
	Numerical exercises	

(*) multiple options are possible

Learning assessment involves an oral test and discussion of a project.

c) Evaluation pattern:

COURSE DETAILS

"INGEGNERIA DEL SUONO"

SSD ING-INF/03

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: FRANCESCO MARIA SACERDOTI
PHONE: 3355824909
EMAIL: FRANCESCOMARIA.SACERDOTI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

No

LEARNING GOALS

The aim of the course is to provide an in-depth knowledge of sound and its processing for the main application areas of interest, such as the creation of multimedia content, concerts and live conferences, music recordings and special effects, audio preparation for films and television programs. The knowledge starts from acoustics and psychoacoustics, up to analog and digital equipment for audio processing.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate to understand the physiology of the human acoustic system and psychoacoustics and, on this basis, to understand the functioning of acoustic perception. In addition, he must know the basics of operation of the main audio equipment to be able to use them and / or design them in the best possible way in the various application areas of interest.

Ability to apply knowledge and understanding

The student must demonstrate that he knows how to reason about acoustic problems and to be able to choose the most suitable technology for solving audio problems.

PROGRAM-SYLLABUS

Audio and Acoustics

Principles of diffusion of audio waves and acoustics. Decibels and level, frequency and wavelength, superposition, impedance. The human ear. Audio monitoring, acoustic radiation

Psychoacoustics

Human perception, the ear and its physiology, audio signal processing in humans. Masking, auditory filters, non-linearity of the ear, phase perception, loudness, mel. Binaural reception, pitch and stamp. The mental process of acoustic perception, localization of sounds. Hall effect, Franssen, Cocktail Party.

Microphones

Types of microphones, classification, spatial response, microphone construction. Proximity effect. Frequency response. Types of cardioids and PZM microphones. Carbon, crystal and ceramic transducers. Dynamic and condenser transducers. Phantom power. Resolution of the nonlinearity of condenser microphones. Ribbon microphones. Measurements on microphones: sensitivity, thermal noise, example of reading characteristics of a microphone. Grounding, polarity, balanced vs. unbalanced, Impedance. Microphone placement. Microphone techniques: Stereo, Mid/Side, 3 to 1, microphone of musical instruments.

Speakers

Using speakers. Construction. Active and passive crossover filters. Speaker types: dynamic, dome, planar, ribbon, horn, piezo, whizzer and coaxial. Boxes: construction and description of construction techniques. Polarity and impedance. Specific. Distortion.

Mixer

Present functions and description of their applicability.

Audio effects

Dynamic processors: compressors and limiters, noise gate and expander, tremolo, autopanner, mazimizer volume. Frequency processors: equalizers, Wah Wah, vibrato, formant, vocoder, talk box. Time-domain processors: echo, reverb, phasers and flangers. Pitch processors. More audio effects.

Other audio equipment

DI Box, patch bay, synthesiser, microphone preamps.

Digital Audio

The digitization of audio. Digitizers. Digital compression. The MIDI protocol. Le DAW Digital Audio Workstations. Software for composition, mixing, mastering, audio processing.

TEACHING MATERIALS

Textbook:

Ballou G. "Handbook for Sound Engineers", Elsevier ISBN: 978-0-240-80969-4

Insights:
Winer E. "The Audio Expert". Routledge ISBN: 978-0-415-78883-0
Alton Everest "The Master Handbook of Acoustics" McGraw Hill, ISBN 978-0-070-19897-5
Douglas S. et al. "Audio Engineering: Know It All" Elsevier ISBN: 978-1-85617-526-5
Miles Huber D. "The MIDI Manual" Elsevier ISBN: 978-0-240-80798-0

TEACHING METHODS

Teaching is provided:
a) 80% with lectures;
b) 20% on laboratory activities for the development of audio software applications to better understand the techniques studied.

The topics of the lectures and seminars are exposed with the help of detailed transparencies, made available to the student in the teaching material through the official website of the teacher.
It is planned to organize a visit to a recording studio of one of the best Italian sound engineers.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	
Other	

The exam is divided into an oral test only, consisting of questions on as many problems exposed to the course on which a reasoning is required by the student for the resolution of particular problems of recording or audio processing.

COURSE DETAILS

"DISTRIBUTED CONTROL AND CYBER PHYSICAL SYSTEMS DESIGN" (ALGORITMI DISTRIBUITI E PROGETTAZIONE DEI SISTEMI DI CONTROLLO SU RETE)

SSD ING-INF/04

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: SABATO MANFREDI
PHONE: 0817683845
EMAIL: SABATO.MANFREDI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

Basic knowledge of closed-loop control systems.

LEARNING GOALS

The course aims at providing students with advanced competences on the analysis and design of Networked Control Systems (NCSs)/Cyber-Physical Systems (CPSs) used for the monitoring and control of distributed and complex processes. Furthermore, it provides students with advanced methodologies for the synthesis of resilient, fault-tolerant and distributed strategies and algorithms, including those based on Artificial Intelligence, for estimation, control and optimization over networks, with application to Industrial, civil and social domains (i.e. Smart Factory, Internet of Thing, Industry 5.0 and Intelligent Automation, Smart City, complex infrastructures and networks, smart grid, distributed computing, swarm of drones, vehicles and aircrafts). The methodologies will be illustrated by the software/hardware design of representative Cyber-Physical Systems.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The course provides students the methodology for the analysis and software/hardware design of modern Networked Control Systems (NCSs)/Cyber-Physical Systems (CPSs). The student needs to show that she/he learned the typical requirements that are peculiar for both hardware and software components of Networked Control Systems (NCSs)/Cyber-Physical Systems (CPSs) used for monitoring/control of industrial/civil processes. The student needs to show ability to know the main design phases of Cyber-Physical Systems and, specifically, of distributed algorithms for estimation, control and optimization over networks. Finally, the student needs to show understanding for the validation and performance evaluation test of distributed algorithms and Cyber-Physical Systems, including the relevance of co-simulation tools to support the design phases.

Applying knowledge and understanding

The student needs to show that she/he is able to formalize the main requirements for a Networked Control System (NCSs)/Cyber-Physical System (CPS) in terms of control system and network performance, including the energy autonomy. Moreover, by starting from such formalized requirements, the student needs to show the ability to design simple algorithms for estimation, control and optimization over networks; and to select the main hardware components. Eventually, the student needs to show the ability to design the validation and performance evaluation test for the developed distributed algorithms and designed Cyber-Physical System; the possibility to exploit simulation tools for such tests should be envisaged.

PROGRAM-SYLLABUS

1. Introduction to Networked Control Systems and Cyber-Physical Systems
   - 1.1 Complex and large-scale distributed systems
   - 1.2 Remote control Systems
   - 1.3 Centralized, decentralized and distributed systems
   - 1.4 Distributed algorithms
   - 1.5 Requirements of Cyber-Physical Systems and distributed algorithms
   - 1.6 Examples of applications
2. Multi-Layer model of Networked Cyber-Physical systems
   - 2.1 “Application” layer
   - 2.2 “Network” layer
   - 2.3 “Physical” layer
   - 2.4 Requirements of Application, Network and Physical layer

3. Distributed algorithms and design of a Cyber-Physical System
   3.1 Multi-agent systems and consensus algorithms
   3.2 Design of control systems at network layer
   3.3 Design of distributed algorithms for load balancing, flow and congestion control
   3.4 Design of control systems at application layer
   3.5 Design of cooperative algorithms for distributed estimation, control and optimization over networks
   3.6 Energy efficiency and energy harvesting in Cyber-Physical Systems. Design of distributed algorithms for energy management
   3.7 AI-based algorithms: Distributed Learning, federated Learning, Learning-based Distributed control
   3.8 Stability analysis, convergence and computational complexity of distributed algorithms
4. Resilience and robustness of Cyber-Physical Systems and Distributed Algorithms
   4.1 Effects of communication delay, packet loss, channel and measurement noise and uncertainty parameters on Cyber-Physical system performance
   4.2 Robust, resilient and fault-tolerant distribute algorithms
5. Distributed algorithms for Cyber-Physical Systems composed of: wireless sensor networks/embedded systems, computer networks and elaboration systems, swarm of drones and vehicles
6. Illustrative applications of software/hardware design methodologies to representative Cyber-Physical Systems for Smart Cities e Smart Factories (Industry 5.0)

TEACHING MATERIALS

[1] Supplementary materials
[2] S. Manfredi, "Multilayer Control of Networked Cyber-Physical Systems. Application to Monitoring, Autonomous and Robot Systems". Advances in Industrial Control, Springer, 2017
[3] A. Bemporad, M. Heemels, M. Vejdemo-Johansson, "Networked Control Systems", Lecture Notes in Control and Information Sciences, Springer, 2010

TEACHING METHODS

The teaching activities will be organized as follows: a) lectures for about 50% of the total hours, b) practical exercise in the classroom based on the simulation tools and/or lab activities for about 50% of the total hours.

EXAMINATION/EVALUATION CRITERIA

(a) Examination procedures:

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

COURSE DETAILS

"BIOINFORMATICA"

SSD ING-INF/06

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: DARIO RIGHELLI
PHONE:
EMAIL: DARIO.RIGHELLI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): II
CFU: 6

PREPARATORY COURSES (if required by the CdS Regulations)

No

PREREQUISITES (IF APPLICABLE)

PROGRAMMING

LEARNING GOALS

The aim of the Bioinformatics course is to provide students with specialized notions related to algorithms for the analysis of genomic data and their possible applications in research in Biomedicine.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The student must demonstrate to be able to describe processes of management and analysis of genomic data in algorithmic form and understand the methodologies adopted in the field of molecular data processing.

Ability to apply knowledge and understanding

The student must demonstrate to be able to consult the main public genomic databases and to describe and apply complex computational procedures in order to extract useful information for biomedical research.

PROGRAM-SYLLABUS

- Introduction to Molecular Biology and biological databases
- Dynamic programming and sequence alignment: Longest Common Subsequence, Edit Distance, Local Alignment, Global Alignment, Substitution Matrices, Multiple Alignments.
- HMM models for modeling genome sequences: Viterbi algorithm, Forward algorithm, Posterior Decoding, Baum-Welsh algorithm. Applications to the classification of genomic sequences. Representation of Multialignments.
- Introduction to Next Generation Sequencing
- Algorithms for Genome Assembly, de Brujin graphs, Hamiltonian paths, and Eulerian paths
- Algorithms for Genome Mapping: Trie for pattern matching, Suffix Trie Tree Matching, Suffix Tree Matching, Suffix Array, Burrows-Wheeler transform and its inverse, pattern matching with BWT
- Molecular phylogeny and phylogenetic trees: UPGMA and Neighbor Joining
- Differential expression analysis, statistical tests, enrichment analysis

TEACHING MATERIALS

- Bioinformatics Algorithms: An Active Learning Approach, by Pavel A. Pevzner and Phillip Compeau https://www.bioinformaticsalgorithms.org/
- Richard Durbin, Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids
- Lecture notes

TEACHING METHODS

The teacher will use:
a) lectures for about 70% of the total hours ;
b) exercises to deepen theoretical aspects for 12 hours
c) seminars for 4 hours.

Teaching will be provided as lectures and computer exercises.

EXAMINATION/EVALUATION CRITERIA

a) **Examination methods:**

The exam is divided into a test	
written and oral	
only written	
oral only	X
Discussion of project paper	X
other	

COURSE DETAILS

"QUANTUM INFORMATION"

SSD ING-INF/03

COURSE NAME: MASTER'S DEGREE IN COMPUTER ENGINEERING

ACADEMIC YEAR: 2025-2026

GENERAL INFORMATIONS- TEACHER REFERENCES

TEACHER: ANGELA SARA CACCIAPUOTI
PHONE: 0817683793
EMAIL: ANGELASARA.CACCIAPUOTI@UNINA.IT

GENERAL INFORMATIONS ABOUT THE COURSE

INTEGRATED COURSE (IF APPLICABLE): N.A.
MODULE (IF APPLICABLE): N.A.
CHANNEL (IF APPLICABLE): N.A.
YEAR OF COURSE (I, II) : I/II
SEMESTER (I, II): I
CFU: 6

PREPARATORY COURSES (if required by the Regulations of the CdS)

No

PREREQUISITES (IF APPLICABLE)

Basic knowledge of linear algebra

LEARNING GOALS

The aim of the course is to provide students with a broad view of information and quantum computing from a communications engineering perspective. Specifically, students will familiarize themselves with the basic elements of quantum information theory, such as qubits, superpositions, quantum measurement, no-cloning and entanglement. Starting from these premises, the main applications will be discussed, including secure communications – analyzing Quantum Key Distribution (QKD) techniques – and quantum communication techniques based on entanglement – such as superdense coding and quantum teleportation. In these scenarios of transmission of classical and quantum information, students will also be provided with the tools to understand the peculiarities of quantum noise compared to classical noise. Students will also acquire the ability to understand the reasons why quantum information processing can enable machine learning and artificial intelligence techniques characterized by performance superior to those guaranteed by classical approaches. Students will have the opportunity to perform simple experiments on a real quantum computer via the IBM Q-Experience platform.
Finally, the course aims to provide the student with the contents and language necessary to allow him to independently deepen the topics covered in the course, to follow in-depth seminars.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The course aims to provide students with the principles and methodological tools necessary to learn and understand the problems that arise with the transmission and processing of quantum information. In particular, the student must acquire the tools to understand the peculiarities of quantum noise compared to classical noise. In addition, the course allows students to understand the similarities and differences between classical and quantum communications, and to grasp the implications and opportunities enabled by those differences.

Ability to apply knowledge and understanding

Students must demonstrate the ability to apply the acquired knowledge and methodological tools to the analysis and design of quantum communication techniques, both in ideal conditions and in the presence of quantum noise. Students must also be able to independently deepen aspects of the topics covered in the course and to follow in-depth seminars, taking advantage of the contents and language provided by the course.

PROGRAM-SYLLABUS

Part I: Fundamentals

- Quantum Information: Qubit vs Bit, Hilbert space, ket-bra notation, Bloch sphere, Multi-qubits systems
    - Lab 1: Introduction to the IBM Quantum (IBM-Q) Experience Platform
    - Lab-2: Visualization of quantum states on IBM-Q
- Quantum Computation: Transformations of quantum states, Theorema of No-Cloning, main quantum gates, quantum measurement
    - Lab-3: Transformations of quantum states on IBM-Q
- Quantum entanglement: Bell states, EPR paradox
    - Lab-4: Generating entanglement on the IBM-Q Platform
- Quantum noise: pure states and mixed states, density operator, decoherence, quantum channel models

Part II: Applications

- Secure Communications: principles of quantum cryptography, BB84 protocol, Ekert-91 protocol, practical implementations
    - Lab-5: BB84 protocol on IBM-Q
- Quantum communications: teleporting protocol, noise effects on teleporting, fidelity

- Lab-6: teleporting protocol on IBM-Q
- Lab-7: state tomography on IBM-Q
- Lab-8: process tomography on IBM-Q

- Quantum information processing for Machine Learning and Artificial Intelligence: simple quantum routines, amplitude amplification and quantum interference, Grover's search learning, distributed quantum processing techniques
  - Lab-9: quantum routines on IBM-Q

TEACHING MATERIALS

Handouts / Slides written by the teacher and available in the dedicated area on docenti.unina.it.

Recommended textbooks:

- Nielsen and Chuang, "Quantum computation and information", Cambridge University Press, 10th Edition, 2020
- Rieffel and Polak, "Quantum Computing: A Gentle Introduction," MIT Press, 2011

TEACHING METHODS

The course is organized by integrating lectures with interactive laboratory sessions. During the course seminars will also be organized inviting experts in the areas of interest and innovative teaching methods, such as flipped classroom and feedback teaching strategies, will be adopted.

EXAMINATION/EVALUATION CRITERIA

a) Examination methods:

The exam is divided into a test	
written and oral	
only written	
oral only	
Discussion of project paper	X
other	

COURSE DETAILS

"ROBOTICS (ROBOTICS LAB)"

SSD ING-INF/04

MASTER DEGREE PROGRAM: COMPUTER ENGINEERING

ACADEMIC YEAR 2024-2025

GENERAL INFORMATION – TEACHER REFERENCES

TEACHER: MARIO SELVAGGIO
PHONE: +39 081 76(83843)
EMAIL: MARIO.SELVAGGIO@UNINA.IT

GENERAL INFORMATION ABOUT THE COURSE

YEAR OF THE DEGREE PROGRAMME: II
SEMESTER: II
CFU: 6

REQUIRED PRELIMINARY COURSES (If mentioned in the course structure "Ordinamento")

PREREQUISITES (IF APPLICABLE)

Basic knowledge of robotics, control, machine learning, programming

LEARNING GOALS

This course will provide the fundamentals tools for building robotic applications using the Robot Operating System (ROS), that provides access to a large set of open-source software modules. Students will learn from partaking in practical activities involving real hardware and software development.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The course path aims to provide students with the software tools for simulating and controlling robotic systems. The Robot Operating System (ROS) will be introduced and used to control robots using basic machine learning and control schemes. The student must demonstrate knowledge in the use of software modules and in the development of machine learning and control algorithms for robot kinematics and dynamics, trajectory planning, vision-based control and mobile robots autonomous navigation.

Applying knowledge and understanding

The student must demonstrate being able to develop software modules and apply them to practical case studies concerning both open-chain robot manipulators and mobile robots. Starting from these, (s)he must demonstrate that (s)he is able to develop machine learning and sensor-based control techniques and validate them using simulated and real robots.

COURSE CONTENT/SYLLABUS

Introduction and course overview

Programming for robotics:

- Preliminary setup: linux operating system, git, docker, C++ programming
- ROS introduction, programming & tools
- ROS robot simulation
- ROS robot motion planning

Motion control of robotic manipulators:

- Kinematics and dynamic control algorithms
- Trajectory planning

Robotic vision:

- Computer vision
- Visual servoing

Control of mobile robots:

- Differential drive mobile robot
- Navigation & SLAM

READINGS/BIBLIOGRAPHY

- L. Joseph, J. Cacace, *Mastering ROS for Robotics Programming*, 2nd Edition, Packt, Birmingham, 2018, ISBN 9781801071024
- Lecture notes available at http://wpage.unina.it/mario.selvaggio/teaching.html

TEACHING METHODS

The teacher will use: a) frontal lessons for about 60% of the total hours, b) classroom exercises for about 40% of the total hours.

EXAMINATION/EVALUATION CRITERIA

a) Exam type:

Exam type	
written and oral	
only written	
only oral	X
project discussion	X
other	

In case of a written exam, questions refer to:	Multiple choice answers	
	Open answers	
	Numerical exercises	

(*) multiple options are possible

Students are admitted to the oral exam after carrying out a project concerning the simulation of the development of control algorithms for robot manipulators and mobile robots. The exam consists of a critical discussion of the paper and in ascertaining the acquisition of the concepts and contents introduced during the lessons.

b) Evaluation pattern:

The development of the project is binding for the purposes of accessing the oral exam. The project and the oral exam each contribute 50% of the final evaluation and, therefore, the development of the project is not sufficient to pass the exam.

COURSE DETAILS

"ROBOTICS (ROBOTIC SYSTEMS)"

SSD ING-INF/04

MASTER DEGREE PROGRAM: COMPUTER ENGINEERING

ACADEMIC YEAR 2024-2025

GENERAL INFORMATION – TEACHER REFERENCES

TEACHER: BRUNO SICILIANO
PHONE: 081 768 3179
EMAIL: BRUNO.SICILIANO@UNINA.IT

GENERAL INFORMATION ABOUT THE COURSE

YEAR OF THE DEGREE PROGRAMME: II
SEMESTER: I
CFU: 6

REQUIRED PRELIMINARY COURSES (If mentioned in the course structure "Ordinamento")

PREREQUISITES (IF APPLICABLE)

Basic knowledge of linear algebra and systems theory

LEARNING GOALS

The course aims to provide the fundamentals of robotic systems.

EXPECTED LEARNING OUTCOMES (DUBLIN DESCRIPTORS)

Knowledge and understanding

The course path aims to provide students with the methodological tools for modeling, planning and control of robotic systems. Robot components, kinematics, differential kinematics, statics, trajectory planning, and basic control schemes are introduced. The student must demonstrate knowledge in the derivation of models and in the validation of algorithms for trajectory planning and kinematic inversion using simulation tools.

Applying knowledge and understanding

The student must demonstrate to be able to derive kinematic and static models and know how to apply them to practical case studies concerning open-chain robot manipulators. Starting from these, (s)he must demonstrate that (s)he is able to design kinematic control schemes and know how to validate them in the Matlab / Simulink® environment.

COURSE CONTENT/SYLLABUS

- Industrial robotics and advanced robotics
- Description and principles of operation of a robot
- Direct kinematics
- Kinematic calibration
- Differential kinematics and Jacobian
- Redundancy and singularities
- Inverse kinematics algorithms
- Kineto-statics duality
- Planning of trajectories in the joint space and in the task space
- Actuators and sensors
- Control unit
- Independent joint control

READINGS/BIBLIOGRAPHY

- B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics – Modelling, Planning and Control, Springer, London, UK, 2009, DOI: 10.1007/978-1-84628-642-1. Italian translation: Robotica – Modellistica, Pianificazione e Controllo, McGraw-Hill Libri Italia, Milano, I, 2008
- B. Siciliano, Robotics Foundations I, MOOC available on the platform www.federica.eu
- B. Siciliano, Robotics Foundations II, MOOC available on the platform www.federica.eu
- Lecture notes available on course website

TEACHING METHODS

The teacher will use: a) frontal lessons for about 70% of the total hours, b) classroom exercises for about 30% of the total hours.

EXAMINATION/EVALUATION CRITERIA

d) Exam type:

Exam type	
written and oral	
only written	
only oral	X
project discussion	X
other	

In case of a written exam, questions refer to:	**Multiple choice answers**	
	Open answers	
	Numerical exercises	

(*) multiple options are possible

Students are admitted to the oral exam after carrying out a design project in Matlab/Simulink® concerning the simulation of inverse kinematics algorithms for robot manipulators. The exam consists of a critical discussion of the paper and in ascertaining the acquisition of the concepts and contents introduced during the lessons.

e) Evaluation pattern:

The development of the project is binding for the purposes of accessing the oral exam. The project and the oral exam each contribute 50% of the final evaluation and, therefore, the development of the project is not sufficient to pass the exam.