UNICUSANO
ROME

Code: ING/INF05 **Credits: 9**
Matter: Software Engineering
Main language of instruction: Italian
Other language of instruction: English

Teaching Staff

Head instructor
Prof. Antonino LONGO MINNOLO - antonino.longo@unicusano.it

Introduction

1. *Objective of the course :*

The Software Engineering Course aims to provide the student with a good knowledge of the principles governing software design. The course proposes the basic concepts relating to the software development process, with particular reference to object-oriented software. Furthermore, the educational objective of the Course is to provide the student with detailed knowledge of the techniques for gathering requirements, drafting specifications, planning, design, implementation, testing, integration and maintenance of the software. The Etivities associated with the Course develop the skills necessary to design and develop software products through the use of suitable software engineering environments.

Objectives

2. *Course Structure:*

- Describe the main requirements collection techniques
- Illustrate the main methodologies for drafting the specifications
- Explain the main software planning and design paradigms
- Illustrate techniques for implementing, testing, integrating and maintaining a software project

Competencies:

A. Knowledge and understanding

At the end of the course, the student will have knowledge of the fundamental concepts and techniques for the collection of requirements, drafting of specifications, planning, design, implementation, integration and maintenance of a software product. Furthermore, the student will acquire a good knowledge of UML (Unified Modeling Language) and of the fundamental aspects of the software development process (with particular reference to object-oriented software). In addition, through the Etivities, students will acquire the ability to deal with the analysis and discussion of concrete case studies.

B. Applying knowledge and understanding

The student will be able to collect and formalize the system requirements, to design even a complex software system and to estimate costs and times; he will also be able to correctly size the various components of a software system and to use UML as a modeling language together with the most common design aid tools.

The Etivities foresee the application of theoretical knowledge to practical problems, such as analysis and discussions on case studies, and group exercises oriented to the implementation of methodologies for software development, to be carried out with the use of suitable development environments and the preparation of a project carried out independently.

C. Making judgements

The student will be able to identify the correct methodologies for the design, implementation and evaluation of computer systems architectures, analyzing various case studies; he will also be able to analyze the data, even limited and incomplete, at his disposal and propose adequate solutions for new problems by integrating the knowledge acquired during the course. Finally, the student will be able to carry out bibliographic searches, to analyze and interpret the relevant sources, in order to analyze the strengths and weaknesses of the proposed solutions, implement risk-driven choices in the planning and implementation of the project and model an IT system from requirements up to to implementation, testing and configuration / installation.

D. Communication skills

The student will be able to describe and hold conversations on issues related to the design and implementation and evaluation of the software product, and to the resolution of typical life cycle management problems, using appropriate terminology.

E. Learning skills.

At the end of the course, the student will be able to independently learn the specific problems relating to the design and implementation of the software.

Syllabus

3. *Programme of the course:*

Subject 1 - Basic concepts and definitions of Software Engineering
Introduction to Software Engineering, Project concepts, activity, resource, task, work product, system, model, document, objectives (goal), requirements, constraints, notations, methods and methodologies. Basic definitions: software products, general characteristics of software products. The qualities of the software. The main stages of development - requirements collection, requirements analysis, system design, executive or object design, implementation, project management, testing, software life cycle. Processes for software development: waterfall model, incremental development; iterative / evolutionary model, prototype model, spiral model, agile model.

Subject 2 – Analysis and specification requirements
Introduction to the collection of requirements. Fundamental concepts: functional requirements, non-functional requirements and pseudo-requirements, levels of description. Main attributes of the specifications (correctness, completeness). Classification of requirements collection activities. Requirements analysis. Identification of actors, scenarios, use cases, relationships between actors and use cases. Identification of the objects of analysis, identification of non-functional requirements. Analysis models - functional, object, dynamic. Requirements analysis and documentation exercise.

Subject 3 – System Design
System design. Introduction, concepts and main activities. Software project. Project methods: top-down, bottom-up approach, structured methods, functional and object oriented strategies. Project documentation. Project quality parameters: cohesion, coupling. Logical architecture project. Principles of object-oriented analysis and design: Classes, Objects, Overloading, Information hiding, Inheritance, Abstract Classes, Interfaces, Dynamic Binding, Exceptions. Exercise for defining and documenting system design.

Subject 4 – Software modeling and design with UML
Generalities on UML (Unified Modeling Language). UML and life cycle. Model requirements with use cases. Diagrams of classes and objects. Sequence diagrams. State diagram. Activity diagram. Components and deployment diagram. Object-oriented modeling.

Subject 5 - Introduction to the Java language
Introduction to the Java language: JVM and JDK. Fundamentals of Object Oriented Programming. The ECLIPSE development environment. Characteristic elements of the Java language. The standard Java libraries. Handling of exceptions. - Handling of errors. Documentation with JavaDoc. I / O in Java. File management. Multithread programming. Interaction with the File System. Networking in Java. Connection to databases: JDBC. Run unit tests with JUnit.

Subject 6 - Verification, validation and testing
The quality control of software products: verification and validation. Introduction to testing; quality control techniques; failure prevention techniques; techniques for fault recognition; fault tolerance techniques. Test concepts: component, failure, error, malfunction, test case, test stub / driver, fix. Testing activities: component inspection; unit testing; integration and system tests. Test planning, test documentation. Exercise on defining, planning and documenting test cases. Run unit tests with JUnit.

Subject 7 - Project management and Design Patterns
Project Management. Fundamental elements of project management Fundamental characteristics of the project. Activities' (ordinary, summary, pivotal). Activity structure. Relationships between activities. The resources (the timetable, the costs). Fixed costs of the project. Project reports. Design patterns. Project management tutorial.

Subject 8 - Agile Design
Agile principles. Agile methods. Scrum methodology. Scrum Team: the Product Owner, the Development Team, the Scrum Master. Scrum Events: Sprint, Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective. The Scrum Artifacts: Product Backlog, Sprint Backlog, Increment. Manage projects with Scrum.

Subject 9 - Software development management
Software configuration management. Configuration item, version, configurations, repository. Use of versioning tools (CVS, SVN, Git). Build, release and branch management.

Evaluation system and criteria
The exam consists in carrying out a written test aimed at ascertaining the ability to analyze and rework the concepts acquired and a series of activities (e-tivity) carried out during the course in virtual classrooms.

The expected learning outcomes about the knowledge of the subject and the ability to apply them are assessed by the written test, while the communication skills, the ability to draw conclusions and the ability to self-learn are assessed in itinere through e-tivities.

Bibliography and resources

4. *Materials to consult:*

Notes written by the instructor are available in English. The notes cover the course contents and examination programme.

5. *Recommended bibliography:*

Suggested readings are:

- I. Sommerville Ingegneria del software 8/Ed. 2007 pp. 848 ISBN 9788871923543
- J. Arlow, I. Neustadt. UML2 e Unified Process - analisi e progettazione Object Oriented, Addison-Wesley
- C. De Sio, Cesari (2014) - Manuale di Java 8. Programmazione orientata agli oggetti con Java standard edition 8 – 1° Edizione - Hoepli – ISBN 8820362910
- Herbert Schildt – Java la guida completa – McGrawill – ISBN 9788838667664