COURSE DESCRIPTION – ACADEMIC YEAR 2024/2025

Course title	**Software Design and Implementation**
Course code	76093
Scientific sector	INF/01
Degree	Master in Software Engineering (LM-18)
Semester	1
Year	1
Credits	12
Modular	Yes

Total lecturing hours	80
Total exercise hours	40
Attendance	Not compulsory. Non-attending students must contact the lecturer at the start of the course to agree on the modalities of the independent study.
Prerequisites	Basic courses in Programming and Software Engineering. Familiarity with UML and software modelling. Familiarity with the basics of object-orientation and automated testing.
Course page	https://ole.unibz.it/

Specific educational objectives	The course belongs to the type "caratterizzanti – discipline informatiche". **Module 1: Requirements Engineering** The course objective is to familiarize students with advanced techniques and tools to elicit, specify and manage software system requirements, aiming to understand both conceptual foundations as well as practical applicability. The students will acquire skills to elicit requirements in various settings and specify them in a way that permits communication with various stakeholders, but also suitable for managing change in software projects. Quality management is specifically introduced. The students are exposed to problem-solving skills that allow requirements engineering in a dynamic, multi-stakeholder setting. **Module 2: Advanced Software Design Techniques** The course objective is to familiarize students with advanced tools and techniques to design a software-based system, understanding in theory and practice their applicability to achieve software requirements and the expected consequences in the implementation of each one. Students will acquire skills and competencies resulting from identifying requirements and implementing them through various design techniques, focusing on improving code maintenance and introducing extensibility in the suitable spots. Design evaluation is approached through the analysis of metrics and visualization techniques. The students are exposed to problem-solving techniques that allow the synthesis of software design solutions satisfying the system's requirements

Module 1	Requirements Engineering
Module code	76093A
Module scientific sector	INF/01
Lecturer	Pahl, Claus
Contact LA	Via Bruno Buozzi 1, Room B1.4.21, Claus.Pahl@unibz.it
Scientific sector of lecturer	INF/01
Teaching language	English
Office hours	During the lecture times, and by arrangement by email
Lecturing Assistant	--
Contact LA	--
Office hours LA	--
Credits	6
Lecturing hours	40
Exercise hours	20
List of topics	• Functional and Non-Functional Requirements • Requirements Engineering Processes • Requirements Elicitation and Analysis • Requirements Specification • Validation of Requirements • Requirements Change
Teaching format	Frontal lectures, exercises; team and/or individual projects.

Module 2	Advanced Software Design Techniques
Module code	76093B
Module scientific sector	ING-INF/05
Lecturer	Martins Guerra, Eduardo
Contact LA	Via Bruno Buozzi 1, Room B1.4.34, eduardo.guerra@unibz.it
Scientific sector of lecturer	ING-INF/05
Teaching language	English
Office hours	Wednesdays from 12h to 14h During the lecture times, and by arrangement by email
Lecturing Assistant	
Contact LA	--
Office hours LA	--
Credits	6
Lecturing hours	40
Exercise hours	20
List of topics	• Design Patterns Application and Interaction • Evolutionary Design Techniques (TDD, BDD, Refactoring) • Domain Modeling (DDD) • Components and Modularization • Framework Development (Extension Points, Reflection, Metadata) • Software Design Evaluation (Code Metrics, Code Smells, Software Visualization)
Teaching format	Frontal lectures, exercises; team and/or individual projects.

Learning outcomes	**Knowledge and understanding** • D1.1 possess solid knowledge of both the fundamentals and the application aspects of the various fundamental areas of computer science; • D1.2 be able to analyse and solve even complex problems in the area of Software Engineering for Information Systems with particular emphasis on the use of empirical evaluation studies, methods, techniques and technologies; • D1.3 have an in-depth knowledge of the scientific method of investigation applied to even complex systems and innovative technologies that support information technology and its applications; • D1.4 have an in-depth knowledge of the principles, structures and use of processing systems for the automation of software systems; • D1.5 know the fundamentals, techniques and methods of design, customisation and implementation of software to support the automation of new-generation information systems for industrial production and company business; • D1.6 understand the elements of corporate and professional culture; • D1.7 know the various fields of application of Software Engineering also with reference to the local, national and international economic-social context; • D1.8 ability to read and understand specialist scientific documentation, such as conference proceedings, articles in scientific journals, technical manuals. **Applying knowledge and understanding** • D2.3 to be able to apply the principles of software engineering to IT and non-IT domains of varying complexity in which software technology is of great importance, such as, for example, in the transport sector or in the medical field; • D2.4 ability to define an innovative technical solution to an application problem that respects technical, functional and organisational constraints and requirements; • D2.5 ability to extend and modify an existing technical solution or formal model in an original way, taking into account changing conditions, requirements and the evolution of technology. **Making judgments** • D3.1 ability to independently select documentation from various sources, including technical books, digital libraries, technical scientific journals, web portals or open source software and hardware tools; • D3.2 ability to plan and re-plan a technical project activity and to carry it out within the defined deadlines and objectives; • D3.3 ability to define work objectives compatible with the available time and resources; • D3.4 ability to reconcile conflicting project objectives, find acceptable compromises within the limits of cost, resources, time, knowledge or risk; • D3.5 be able to work with broad autonomy, including taking responsibility for projects and structures.

	Communication skills • D4.1 ability to present the contents of a scientific/technical report in a set time in front of an audience, including non-specialists; • D4.2 ability to structure and draft scientific and technical descriptive documentation of project activities; • D4.3 ability to co-ordinate the work of a project team and to identify activities aimed at achieving the project objectives; • D4.4 ability to prepare and deliver presentations with technical content in English; • D4.5 ability to interact and collaborate in the realisation of a project or research with peers and experts; • D4.6 ability to carry out research and projects in a working group; • D4.7 ability to synthesise knowledge gained from reading and studying scientific and technical documentation and to prepare reports and presentations. **Learning skills:** • D5.1 ability to independently extend the knowledge acquired during the course of study by reading and understanding scientific and technical documentation in English; • D5.2 ability to independently keep up to date with developments in the most important fields of information technology; • D5.3 in the context of a problem solving activity, ability to extend even incomplete knowledge with regard to the final objective of the project.

Assessment	**Module 1: Requirements Engineering** The assessment is based on the lab assessment and the final written exam. The lab assessment is composed practical activities that can be performed by the students during the course. The final written exam evaluates the students' understanding of the theoretical backgrounds and the ability of solving problems. The student should achieve at least 50% of the lab grade to do the final exam. **Module 2: Advanced Software Design Techniques** The assessment is based on the lab assessment and the final written exam. The lab assessment is composed of practical activities that can be performed by the students during the course. The final written exam evaluates the students' understanding of the theoretical background and the ability to solve problems. The student should achieve at least 50% of the lab grade to do the final exam.
Assessment language	English
Assessment typology	Collegial commission
Evaluation criteria and criteria for awarding marks	**Module 1: Requirements Engineering** For attending students, the grade is calculated based on (i) the lab assessment (50% weight) and (ii) the written final exam (50% weight). For non-attending students, they should follow the delivery schedule for the lab assessments, the grade is calculated the same way.

	Module 2: Advanced Software Design Techniques For attending students, the grade is calculated based on (i) the lab assessment (50% weight) and (ii) the written final exam (50% weight). For non-attending students, they should follow the delivery schedule for the lab assessments, the grade is calculated the same way. A student needs to be approved in both modules to be approved in the course. The final grade is the average value of the grades from both modules.

Required readings	• Subject Librarian: David Gebhardi, David.Gebhardi@unibz.it

Supplementary readings	• Sommerville, I. (2015). Software Engineering. 10th Edition. Pearson. • Laplante, P.A., and Kassab, M.H. (2022). Requirements Engineering for Software and Systems. CRC Press. • Johnson, R., & Vlissides, J. (1995). Design patterns. Elements of Reusable Object-Oriented Software Addison-Wesley, Reading. • Beck, K. (2003). Test-driven development: by example. Addison-Wesley Professional. • Fowler, M. (2018). Refactoring: improving the design of existing code. Addison-Wesley Professional. • Evans, E., & Evans, E. J. (2004). Domain-driven design: tackling complexity in the heart of software. Addison-Wesley Professional. • Lanza, M., & Marinescu, R. (2007). Object-oriented metrics in practice: using software metrics to characterize, evaluate, and improve the design of object-oriented systems. Springer Science & Business Media. • Open educational resources, representing alternative or supplementary materials, shall be linked to the course website.
Software used	Software Modelling (e.g., Argo UML, Papyrus, StarUML, draw.io), Java JDK, Java Programming IDE (e. g. Eclipse, Intellij)