Fakultät für Informatik
Facoltà di Scienze e Tecnologie informatiche
Faculty of Computer Science

SYLLABUS
COURSE DESCRIPTION

COURSE TITLE	Software Engineering
COURSE CODE	75037
SCIENTIFIC SECTOR	ING-INF/05
DEGREE	Bachelor in Computer Science and Engineering
SEMESTER	2nd Semester
YEAR	2nd year
CREDITS	8

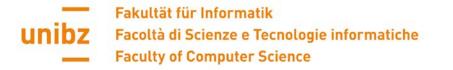|	
TOTAL LECTURING HOURS	48
TOTAL LAB HOURS	24
PREREQUISITES	Introduction to Programming, Advanced Programming
COURSE PAGE	https://ole.unibz.it/

SPECIFIC EDUCATIONAL OBJECTIVES	• Type of course: "caratterizzante" for L-31 and L-08 • Scientific area: "discipline informatiche" for L-31 and "ingegneria informatica" for L-8 The course introduces the state-of-the-art in software engineering. It aims to demonstrate how this is transferred into practically applicable knowledge and skills for software development.

LECTURER	Claus Pahl Piazza Domenicani 3, Office 1.11, Claus.Pahl@unibz.it, +39 0471 016 177
SCIENTIFIC SECTOR OF THE LECTURER	INF/01
TEACHING LANGUAGE	English
OFFICE HOURS	During the lecture times, and Monday 14:00-16:00. Faculty of CS, Piazza Domenicani 3, Office 1.11
TEACHING ASSISTANT	Claus Pahl: Claus.Pahl@unibz.it Nabil El Ioini: Nabil.ElIoini@stud-inf.unibz.it
OFFICE HOURS	Nabil El Ioini: Mondays, 16:00-18:00, Office 1.08, Piazza Domenicani 3;

LIST OF TOPICS COVERED	• Software life-cycle • Software processes • Requirements engineering • System modelling (UML) • Software construction (from Java to C++) • Principles of software testing • Managing software projects • Software evolution
TEACHING FORMAT	Frontal lectures, exercises, projects.

LEARNING OUTCOMES	**Knowledge and understanding** • Know in detail principles, techniques and methods of planning, designing, developing and maintaining software; **Applying knowledge and understanding** • be able to apply the own knowledge to the analysis, design, development and testing of information systems which satisfy given requirements; • be able to solve typical problems in computer science, such as the definition of requirements, the analysis of possible methods for a solution, the selection of methods and tools as well as their application; • be able to evaluate the quality of information systems and to identify critical aspects; • be able to apply the own knowledge in different working contexts; **Making judgments** • be able to take the responsibility for software development projects **Communication skills** • be able to explain a project activity or a scientific study, also to non-experts • be able to work in teams to implement software systems **Learning skills** • have acquired learning capabilities that enable them to carry out project activities in companies, public institutions or in distributed development communities • be able to learn cutting edge IT technologies and their strengths and limitations

ASSESSMENT	• Written and project work: written exam with verification questions and written project report done in groups • In case of a positive mark the project will count for all 3 regular exam sessions. • Projects have to be submitted BEFORE the final exam at the end of the semester, otherwise the exam cannot be registered.
ASSESSMENT LANGUAGE	English
EVALUATION	Weighting of parts:

CRITERIA AND CRITERIA FOR AWARDING MARKS	70% written 30% exercises/project Criteria Relevant for assessment of project and exam: - clarity of answers, - mastery of language, - skills in critical thinking - ability to summarize, evaluate, and establish relationships between topics, - technical competence Relevant for project assessment: - ability to work in a team, - creativity, - development skills

REQUIRED READINGS	The course will be based on lecture notes
SUPPLEMENTARY READINGS	I. Sommerville. Software Engineering. Addison Wesley.
SOFTWARE USED	Software development IDEs (e.g., Eclipse, NetBeans), Management tools (subversion), Modelling (e.g. Argo UML), Testing (e.g. JUnit)