COURSE OUTLINE

(1) GENERAL

SCHOOL	SCHOOL OF ENGINEERING		
ACADEMIC UNIT	Department of Informatics and Computer Engineering		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	ICE-6001	**SEMESTER**	**6th**
COURSE TITLE	Software Engineering		
INDEPENDENT TEACHING ACTIVITIES if credits are awarded for separate components of the course, e.g. lectures, laboratory exercises, etc. If the credits are awarded for the whole of the course, give the weekly teaching hours and the total credits		**WEEKLY TEACHING HOURS**	**CREDITS**
Lectures		3	
Laboratories		1	
		4	5
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).			
COURSE TYPE general background, special background, specialised general knowledge, skills development	Scientific Area, Skills Development		
PREREQUISITE COURSES:			
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek		
IS THE COURSE OFFERED TO ERASMUS STUDENTS	No		
COURSE WEBSITE (URL)	https://eclass.uniwa.gr/courses/CS212/		

(2) LEARNING OUTCOMES

Learning outcomes

The course learning outcomes, specific knowledge, skills and competences of an appropriate level, which the students will acquire with the successful completion of the course are described.

Consult Appendix A

- Description of the level of learning outcomes for each qualifications cycle, according to the Qualifications Framework of the European Higher Education Area
- Descriptors for Levels 6, 7 & 8 of the European Qualifications Framework for Lifelong Learning and Appendix B
- Guidelines for writing Learning Outcomes

The course aims to enable students to master the basic principles and fundamental characteristics of software product development, as an industrial production process subject to quality control, to master the basic knowledge of structured and object-oriented software development methodologies & to apply software development methodologies using appropriate software tools (Businessmodeling, CASE).

The main learning objectives of the course are located in:

Assimilation of the basic concepts of software development,

- Knowledge of the main software life cycle models
- Knowledge and application of structured software development methodology (e.g. SSADM)
- Knowledge and application of the object-oriented UML methodology
- Knowledge and application of horizontal (scale-out) and vertical (scale-up) principles
- software systems scaling
- Understanding of the fundamental quality characteristics of IT systems and adopting the principles of good software design
- Knowledge and application of software control, acceptance, maintenance and release management processes
- Knowledge and application of the basic principles of software project management
- Knowledge and application of the functional capabilities of CASE tools and acquiring the ability to use them in software development

- Ability to identify “software development process” maturity through the CMM model
- Ability to identify and solve problems that arise during software design
- Ability to develop original software based on the best practices found in the literature

General Competences

Taking into consideration the general competences that the degree-holder must acquire (as these appear in the Diploma Supplement and appear below), at which of the following does the course aim?

Search for, analysis and synthesis of data and information, with the use of the necessary technology	Project planning and management
	Respect for difference and multiculturalism
	Respect for the natural environment
Adapting to new situations	Showing social, professional and ethical responsibility and sensitivity to gender issues
Decision-making	
Working independently	Criticism and self-criticism
Team work	Production of free, creative and inductive thinking
Working in an international environment	……
Working in an interdisciplinary environment	Others…
	…….
Production of new research ideas	

- Search, analysis and synthesis of data and information, using the necessary technologies
- Autonomous work
- Teamwork
- Planning and Project Management
- Prduction of new research ideas

(3) SYLLABUS

- Historical review, fundamentals of software technology, n
- Software development as an industrial process.
- Software life cycle, life cycle models.
- Software development methodologies (data flow oriented, data structure oriented- & object oriented software development etc.). Techniques and Examples
- Strategies for horizontal and vertical scaling of systems
- Introduction to UML (Use case design, Class diagrams, code representations in JAVA/C++, Sequence diagrams, Activity Diagrams)
- Software design architectures – Coupling, coherence, fan-in, fan-out etc.
- CASE technology (Architectures, how to import and utilize it, presentation of software development environments, etc.).
- Software correctness checking, software acceptance criteria
- Software Maintenance & Software Version Management
- Software Project Management
- Software reuse, reverse engineering.
- Basic software quality assurance concepts
- The maturity of the software development process (Humphrey's CMM model)
- Use of CASE technology

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY Face-to-face, Distance learning, etc.	Face-to-face
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY Use of ICT in teaching, laboratory education, communication with students	▪ Use of electronic material to support courses (slides and material in electronic format) ▪ Use of code generators (specialized CASE tools for software design and production using object oriented technologies & use of business modeling tools). ▪ • Learning process support through e-class electronic platform
TEACHING METHODS The manner and methods of teaching are described in detail. Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, placements, clinical practice, art workshop, interactive teaching, educational visits, project, essay writing, artistic creativity, etc. The student's study hours for each learning activity are given as well as the hours of non-directed study according to the principles of the ECTS	**Activity** / **Semester workload** Lectures: 39 Laboratory practice: 13 Assignments: 21 Independent personal study: 52 Course total: **125**
STUDENT PERFORMANCE EVALUATION Description of the evaluation procedure Language of evaluation, methods of evaluation, summative or conclusive, multiple choice questionnaires, short-answer questions, open-ended questions, problem solving, written work, essay/report, oral examination, public presentation, laboratory work, clinical examination of patient, art interpretation, other Specifically-defined evaluation criteria are given, and if and where they are accessible to students.	I. Written final exams (60%) that includes: – Theory questions – Problem solving – Comparative evaluation of theory elements II. Lab assignment (40%)

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography (Greek)

1. Τεχνολογία Λογισμικού, Γιακουμάκης, Διαμαντίδης, εκδ. Σταμούλης
2. Βασικές Αρχές Τεχνολογίας Λογισμικού I. Sommerville, Κλειδάριθμος (Μετάφραση στα Ελληνικά)
3. Τεχνολογία Λογισμικού, Θεωρία & Πράξη Τόμος I, S. Pfleeger, Κλειδάριθμος (Μετάφραση στα Ελληνικά)
4. SWEBOK. IEEE Computer Society. 2001. In english. Free download: http://www.swebok.org/
5. Σημειώσεις εργαστηρίου για τη χρήση του εργαλείου CASE "RationalRose"& του business modeling tool ADONIS – community Edition
6. Τεχνολογία Λογισμικού, Βασίλειος Βεσκούκης, ΣΕΑΒ 2015, Ηλεκτρονική έκδοση
7. Διαλέξεις Μαθήματος «Τεχνολογία Λογισμικού», Ιωάννης Χάλαρης, e-class