Università degli Studi di L'Aquila - Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Course catalogue

Page compiled on 23/10/2019

Programme of Integrated course "Software Architectures & Quality Engineering"

This course is composed of 2 Modules: 1) Software Architectures, 2) Software Quality Engineering

Programme of Module "Software Architectures"

- Code: DT0223
- Type of course unit: Compulsory (Master Degree in Computer Science curriculum NEDAS), Compulsory (Master Degree in Computer Science curriculum UBIDIS)
- Level of course unit: Postgraduate Degrees
- Semester: 1

Number of ects credits: (Master Degree in Computer Science) 6 (workload 150 hours)

Teachers: Henry Muccini (henry.muccini@di.univaq.it)

1	**Course objectives**	Introducing the students to Software Architectures and to the architecting process.
2	**Course content and learning outcomes (dublin descriptors)**	Topics of the module include: • Components and Connectors • Architectural Styles • Architectural Views and Viewpoints • Architecture Descriptions and Architecture Description languages • Architecture Design Decisions • Architecting Situational Aware Applications On successful completion of this module, the student should : • KNOWLEDGE: This course introduces advanced concepts on Software Architecture. The first part of this course will provide advanced basic and advanced knowledge on software architecture, together with examples, and with a specific focus on architecture description language, and multi-view modeling. The second part will focus on architectural design decisions, architectural patterns, and architecture for adaptive systems. An objective is also to gain familiarity with software languages and tools which make easier the specification of component-based systems and architectures ABILITY (ability to do): From the perspective of the "ability students will gain", the main objective of this course is to acquire a good knowledge on both theory and practice of Software Architecture and their usage in practice. At the end of this course, students will be able to correctly model a Software Architecture by using the appropriate tools. Through projects, students will practice the theoretical concepts previously described. BEHAVIOR (ability to be): at the end of the learning process, the students will be conscious of how architectural choices impact on the quality of the developed software system.
3	**Course prerequisites**	Basics on Software Engineering
4	**Teaching methodsand language**	Readings, slides, textbook **Language**: English **Reference textbooks** • Len Bass, Paul Clements, and Rick Kazman, ***Software Architecture in Practice (3rd Edition).*** . Addison Wesley Professional. 2012.
5	**Assessment methods**	project and oral exam

Programme of Module "Software Quality Engineering"

- Code: DT0204
- Type of course unit: Compulsory (Master Degree in Computer Science curriculum GSEEM), Elective (Master Degree in Computer Science curriculum General), Elective (Master Degree in Computer Science curriculum

NEDAS), Compulsory (Master Degree in Computer Science curriculum SEAS), Elective (Master Degree in Computer Science curriculum UBIDIS)
- Level of course unit: Postgraduate Degrees
- Semester: 1

Number of ects credits: (Master Degree in Computer Science) 6 (workload 150 hours)

Teachers: Vittorio Cortellessa (Vittorio.Cortellessa@univaq.it)

1	**Course objectives**	This course aims to more deeply explore some of the concepts covered during the course of Software Engineering Basics and to introduce new concepts. In particular, this course deals with: non-functional properties of software architecture (such as reliability and performance), with a particular emphasis on their quantitative assessment. It intends to let the student acquiring not only software modeling and analysis skills, through the use of tools that support these activities, but it also aims at developing the student ability to adapt to different tools and to interpret the results that these tools can offer.
2	**Course content and learning outcomes (dublin descriptors)**	Topics of the module include: • Software Architectures • Model-Driven Engineering • UML profiling • Non-functional Validation of Software • Dependability (definitions) • Performance Analysis On successful completion of this module, the student should : • o Be aware of potential of software models as primary artifacts in the whole software engineering process. o Be able to use UML profiles for tailoring software architectures to specific domains. o Be experienced in the integration of multiple tools for the development and analysis of software systems. o Be able to design models that reflect abstract architectures of software systems. o Be able to analyze a software architecture from a non-functional viewpoint, with particular emphasis for reliability and performance aspects. o Be able to identify and define the computing requirements appropriate to its solution. o Have effectively worked on team to deliver some group homework.
3	**Course prerequisites**	This is an advanced course in the area of Software Engineering, so it is assumed that students have already taken a course of Software Engineering foundations. UML knowledge is also required.
4	**Teaching methodsand language**	The course language is English. It includes 48 hours of frontal lectures, which are partitioned in theory, exercises and homework discussions. **Language**: English **Reference textbooks** • Vittorio Cortellessa, Antinisca Di Marco, Paola Inverardi, ***Model-based Software Performance Analysis***. Springer. • Connie U. Smith, Lloyd Williams, ***Performance Solutions***. Addison Wesley. • Ian Sommerville, ***Software Engineering***. Addison Wesley.
5	**Assessment methods**	There is no formal pre-assessment, apart from Course pre-requisites. Fulfilment of such pre-requisites is verified by formative assessment. The formative assessment is performed via interactions between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: Questioning and discussion, by means of open oral questions to the class or to single students; Summative Assessment Group project followed by an optional oral exam. The group project is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents; (2) verification of skills in understanding and solving significant problems, and in explaining the proposed solutions, (3) capability of collaborative work. This is aimed at verifying the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of a technical/mathematical language; the clarity and completeness of explanations. The oral exam will occur within one week from the project delivery and will typically cover the areas of the project that need clarification plus additional subjects proposed by the

		teacher and included in the syllabus. The oral test takes place for all students. Assessment breakdown: 100% end-of-semester summative assessment.