Teaching Guide				
Identifying Data				2020/21
Subject (*)	Software Architecture		Code	614G01221
Study programme	Grao en Enxeñaría Informática			
Descriptors				
Cycle	Period	Year	Type	Credits
Graduate	1st four-month period	Adaptation Course for Technical Engineers	Obligatory	6
Language	Spanish			
Teaching method	Face-to-face			
Prerequisites				
Department	Computación			
Coordinador		E-mail		
Lecturers		E-mail		
Web	guiadocente.udc.es/guia_docent/index.php?centre=614&ensenyament=614G01&assignatura=614G01026&any_academic=2017_18&am			
General description	This subject is intended to master current Software Engineering solutions for the design of applications and systems, in the architectural level. This involves: - Knowledge of the most typical software architectures and their properties; - Study of non-functional requirements and their relationship to software architecture; - Development and/or study of actual systems.			
Contingency plan	1. Modifications to the contents 2. Methodologies *Teaching methodologies that are maintained *Teaching methodologies that are modified 3. Mechanisms for personalized attention to students 4. Modifications in the evaluation *Evaluation observations: 5. Modifications to the bibliography or webgraphy			

Study programme competences / results	
Code	Study programme competences / results
A25	Capacidade para desenvolver, manter e avaliar servizos e sistemas sóftware que satisfagan todos os requisitos do usuario e se comporten de forma fiable e eficiente, sexan accesibles de desenvolver e manter, e cumpran normas de calidade, aplicando as teorías, principios, métodos e prácticas da enxeñaría do sóftware.
A27	Capacidade de dar solución a problemas de integración en función das estratexias, estándares e tecnoloxías dispoñibles.
A28	Capacidade de identificar e analizar problemas, e deseñar, desenvolver, implementar, verificar e documentar solucións sóftware sobre a base dun coñecemento adecuado das teorías, modelos e técnicas actuais.
B1	Capacidade de resolución de problemas
B2	Traballo en equipo
B3	Capacidade de análise e síntese
B4	Capacidade para organizar e planificar

C3	Utilizar as ferramentas básicas das tecnoloxías da información e as comunicacións (TIC) necesarias para o exercicio da súa profesión e para a aprendizaxe ao longo da súa vida.
C4	Desenvolverse para o exercicio dunha cidadanía aberta, culta, crítica, comprometida, democrática e solidaria, capaz de analizar a realidade, diagnosticar problemas, formular e implantar solucións baseadas no coñecemento e orientadas ao ben común.
C6	Valorar criticamente o coñecemento, a tecnoloxía e a información dispoñible para resolver os problemas cos que deben enfrontarse.

Learning outcomes			
Learning outcomes	Study programme competences / results		
Learn Software Engineering concepts and techniques.	A25		
Understand and identify the typical problems of software architectures and their contexts.	A25 A27 A28	B2 B3	C4 C6
Define and document specifications, models, and architectural components of an application, according to their requirements, so as to favour their maintenance and extensibility.	A28	B1 B2 B3 B4	
Proficient use of modeling languages.	A28		
Use specific tools for defining and building applications.			C3
Validate the architecture of a system against its requirements.	A25		
Synthesize success stories.	A25	B3	C4 C6

Contents	
Topic	Sub-topic
Concept of software architecture	Definition of software architecture Structures and views - Notation -- UML -- IEEE Standard 1471 - Tools Life and business cycle of software architecture
Reference models and architectures	Quality indicators in software architecture Types of architectures - Layered architecture - Repository architecture - Client/server architecture (service-oriented) - 'Pipe and filter' architecture (component-based) - Distributed architectures -- Master/slave architectures -- Multilayered client/server architectures -- P2P architectures - Other architectures -- Embedded systems -- Aspect-oriented systems

Component design and integration. Architectural patterns	Design strategies Architectural Patterns - Patterns for service access and configuration - Patterns for event management - Synchronization patterns - Distribution patterns - Concurrency patterns Reuse - Legacy and COTS systems - Integration styles -- File transfer -- Data sources sharing -- Remote procedure invocation -- Message passing System reconstruction / re-engineering
Traceability and integration testing	Integration process Verification and integration testing - Functional tests - Non-functional tests Validation and Usability

Planning				
Methodologies / tests	Competencies / Results	Teaching hours (in-person & virtual)	Student?s personal work hours	Total hours
Guest lecture / keynote speech	B3	21	21	42
Document analysis	B3 B4 C3	0	7	7
Directed discussion	A28 B1 B3 C6	7.5	15	22.5
Laboratory practice	A25 A27 A28 B1 B2 B4 C4 C6	15	30	45
Supervised projects	A27 A28 B1 B3 B4 C3 C6	1.5	15	16.5
Objective test	A27 A28 B1 B3 C6	3	9	12
Personalized attention		5	0	5
(*)The information in the planning table is for guidance only and does not take into account the heterogeneity of the students.				

Methodologies	
Methodologies	Description
Guest lecture / keynote speech	Lectures in which the notions and concepts of the field are presented, using different kinds of resources such as board, slides, or material provided beforehand by the teacher by means of a virtual platform (Moodle).
Document analysis	Reading and understanding task for the student, in which they will manage different resources provided or pointed to. Materials will be selected to promote a better understanding of lectures, to generate debate during discussion sessions, or to assist in carrying out practical (un)supervised work.
Directed discussion	Constructive debate, led by the teacher but participated by the whole class group, on different issues presented in lectures. The aim of these debates is to deepen the understanding and acquisition of theoretical concepts, and the development of critical and analytical skills.
Laboratory practice	Small projects designed so that the students can put in practice the theoretical knowledge as they acquire it. These projects wil be dimensioned to be undertaken by groups of students. The size of these gropus will be determined depending on the number of students enrolled in the course.
Supervised projects	Specific report or essays to be developed by students, either in groups or individually. These reports will be presented either at small group sessions or during personalized tutoring sessions.

Objective test	Final examination in which students must prove the knowledge they have acquired. Students are expected to show their skills both on a theoretical level (by answering questions similar to those posed during lectures and discussion sessions), and a practical level (by solving problems and exercises similar to those proposed during lab sessions and small projects).

Personalized attention	
Methodologies	**Description**
Laboratory practice Supervised projects	The personalized attention to students involvese not only the well-known tutoring sessions, but also the following actions: - Guidance and monitoring of the work done in the projects/essays/reports and other practices. - Evaluation of the involvement and participation in discussion sessions.

Assessment			
Methodologies	**Competencies / Results**	**Description**	**Qualification**
Laboratory practice	A25 A27 A28 B1 B2 B4 C4 C6	Evaluation of the practices (small projects). Even though these practices are conducted in groups, two components are considered in the assessment of a student's work: - Assessment of group work, which takes into account the degree of coordination and collaboration among its members. - Personal assessment, which evaluates the specific contribution of one student to the group. The aspects that will be considered to evaluate these projects are: - Accuracy in achieving the objectives using the proposed techniques. - Understanding of the concepts involved. - Originality of the proposals. - Responsibility in delivering the project results in due time, as well as proper use of the established delivery means.	40
Objective test	A27 A28 B1 B3 C6	Written test divided into two parts: theoretical questions, and modeling of a problem.	40
Supervised projects	A27 A28 B1 B3 B4 C3 C6	The following aspects will be evaluated: - Knowledge and understanding of presented contents. - Knowledge and understanding of the theoretical and practical concepts of the subject involved.	20

Assessment comments
Students will need to show balance in their performance on the final examination and the lab practices (group projects). A balance of at least 50% of the corresponding qualification weight will be required on both aspects. In the second chance evaluation, the objective test can include a laboratory evaluation for those people which do not reach 50% of the laboratory practice grade during the semester. In compliance with the academic rules at UDC that apply to part-time students, physical presence in the classroom/laboratory will not be regarded as qualification element. That is to say, students may officially apply to be dismissed from attending lectures and laboratory practices. All in all, part-time students will still need to comply with deadlines established for supervised projects and laboratory projects, since these will be announced in the subject webpage, and the projects will always be handed-in electronically.

Sources of information

Basic	- Sommerville, Ian (2011). Ingeniería de software. Addison Wesley - Schmidt, Douglas [et al.] (2000). Pattern-oriented software architecture. John Wiley & Sons - Braude, Eric J. (2001). Software engineering an object-oriented perspective. John Wiley & Sons - Fowler, Martin (2003). Patterns of enterprise application architecture. Addison-Wesley - Bass, Len [et al.] (2003). Software architecture in practice. Addison-Wesley - Clements, Paul [et al.] (2003). Documenting software architectures : views and beyond. Addison-Wesley - Hohpe, Gregor (2004). Enterprise integration patterns designing, building and deploying messaging solutions. Addison-Wesley
Complementary	

Recommendations
Subjects that it is recommended to have taken before
Software Design/614G01015 Software Process/614G01019 Internet and Distributed Systems/614G01023
Subjects that are recommended to be taken simultaneously
Requirements Engineering/614G01027 Quality Assurance/614G01028
Subjects that continue the syllabus
Development Frameworks/614G01052 Software Verification and Validation/614G01053 Development Tools/614G01054
Other comments

(*)The teaching guide is the document in which the URV publishes the information about all its courses. It is a public document and cannot be modified. Only in exceptional cases can it be revised by the competent agent or duly revised so that it is in line with current legislation.