



Teaching Guide				
Identifying Data				2013/14
Subject (*)	Arquitectura do Software	Code	614G01026	
Study programme	Grao en Enxeñaría Informática			
Descriptors				
Cycle	Period	Year	Type	Credits
Graduate	2nd four-month period	Third	Obligatoria	6
Language	Spanish			
Prerequisites				
Department	Computación			
Coordinador	Castro Souto, Laura Milagros	E-mail	laura.milagros.castro.souto@udc.es	
Lecturers	Cabrero Souto, David Castro Souto, Laura Milagros Valderruten Vidal, Alberto	E-mail	david.cabrero@udc.es laura.milagros.castro.souto@udc.es alberto.valderruten@udc.es	
Web	campusvirtual.udc.es			
General description	Esta materia busca dominar as alternativas actuais da enxeñaría do software para ol deseño de aplicacións e sistemas a nivel de arquitectura: ? Coñecendo as arquitecturas máis típicas e as súas características; ? Estudando os requerimentos non funcionales dos sistemas e a súa relación coa arquitectura; e ? Desenvolvendo e/ou estudando sistemas reais.			

Study programme competences	
Code	Study programme competences
A5	Coñecemento da estrutura, organización, funcionamento e interconexión dos sistemas informáticos, os fundamentos da súa programación e a súa aplicación para a resolución de problemas propios da enxeñaría.
A7	Capacidade para deseñar, desenvolver, seleccionar e avaliar aplicacións e sistemas informáticos que aseguren a súa fiabilidade, seguranza e calidade, conforme a principios éticos e á lexislación e normativa vixente.
A8	Capacidade para planificar, concibir, despregar e dirixir proxectos, servizos e sistemas informáticos en todos os ámbitos, liderando a súa posta en marcha e a súa mellora continua e valorando o seu impacto económico e social.
A9	Capacidade para comprender a importancia da negociación, os hábitos de traballo efectivos, o liderado e as habilidades de comunicación en todos os contornos de desenvolvemento de sóftware
A10	Capacidade para elaborar o prego de condicións técnicas dunha instalación informática que cumpra os estándares e as normativas vixentes.
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.
A29	Capacidade de identificar, avaliar e xestionar os riscos potencias asociados que se puideren presentar.
A33	Capacidade de analizar e avaliar arquitecturas de computadores, incluíndo plataformas paralelas e distribuídas, así como desenvolver e optimizar sóftware para elas
A48	Capacidade para participar activamente na especificación, deseño, implementación e mantemento dos sistemas de información e comunicación.
A53	Capacidade para seleccionar, deseñar, despregar, integrar, avaliar, construír, xestionar, explotar e manter as tecnoloxías de hardware, sóftware e redes dentro dos parámetros de custo e calidade adecuados.
B1	Capacidade de resolución de problemas
B2	Traballo en equipo
B3	Capacidade de análise e síntese
B4	Capacidade para organizar e planificar



B5	Habilidades de xestión da información
B6	Toma de decisións
B7	Preocupación pola calidade
B8	Capacidade de traballar nun equipo interdisciplinar
B9	Capacidade para xerar novas ideas (creatividade)
C1	Expresarse correctamente, tanto de forma oral coma escrita, nas linguas oficiais da comunidade autónoma.
C2	Dominar a expresión e a comprensión de forma oral e escrita dun idioma estranxeiro.
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.
C7	Asumir como profesional e cidadán a importancia da aprendizaxe ao longo da vida.
C8	Valorar a importancia que ten a investigación, a innovación e o desenvolvemento tecnolóxico no avance socioeconómico e cultural da sociedade.

Learning outcomes			
Subject competencies (Learning outcomes)	Study programme competences		
Learn Software Engineering concepts and techniques.	A5		
Understand and identify the typical problems of software architectures and their contexts.	A10	B2	C1
	A25	B3	C2
	A27	B5	C4
	A28	B7	C6
	A29	B8	C7
	A48	B9	C8
Define and document specifications, models, and architectural components of an application, according to their requirements, so as to favour their maintenance and extensibility.	A7	B1	
	A8	B2	
	A9	B3	
	A33	B4	
		B5	
		B6	
		B7	
		B8	
		B9	
Proficient use of modeling languages.	A28		
Use specific tools for defining and building applications.			C3
Validate the architecture of a system against its requirements.	A7	B7	
	A25		
	A53		
Synthesize success stories.	A7	B3	C1
	A25	B5	C2
	A29		C4
			C6
			C7
			C8

Contents	
Topic	Sub-topic



Concept of software architecture	<p>Definition of software architecture</p> <p>Structures and views</p> <ul style="list-style-type: none"> <li>- Notation</li> <li>-- UML</li> <li>-- IEEE Standard 1471</li> <li>- Tools</li> </ul> <p>Life and business cycle of software architecture</p>
Reference models and architectures	<p>Quality indicators in software architecture</p> <p>Types of architectures</p> <ul style="list-style-type: none"> <li>- Layered architecture</li> <li>- Architecture repository</li> <li>- Client/server architecture (service-oriented)</li> <li>- 'Pipe and filter' architecture (component-based)</li> <li>- Distributed architectures <ul style="list-style-type: none"> <li>-- Master/slave architectures</li> <li>-- Multilayered client/server architectures</li> <li>-- P2P architectures</li> </ul> </li> <li>- Other architectures <ul style="list-style-type: none"> <li>-- Embedded systems</li> <li>-- Aspect-oriented systems</li> </ul> </li> </ul>
Component design and integration. Architectural patterns	<p>Design strategies</p> <p>Architectural Patterns</p> <ul style="list-style-type: none"> <li>- Patterns for service access and configuration</li> <li>- Patterns for event management</li> <li>- Synchronization Patterns</li> <li>- Distribution patterns</li> <li>- Patterns for competitiveness</li> </ul> <p>Reuse</p> <ul style="list-style-type: none"> <li>- Legacy and COTS systems</li> <li>- Integration styles <ul style="list-style-type: none"> <li>-- File transfer</li> <li>-- Data sources sharing</li> <li>-- Remote procedure invocation</li> <li>-- Message passing</li> </ul> </li> </ul> <p>System reconstruction / re-engineering</p>
Traceability and integration testing	<p>Integration process</p> <p>Verification and integration testing</p> <ul style="list-style-type: none"> <li>- Functional tests</li> <li>- Non-functional tests</li> </ul> <p>Validation and Usability</p>

Planning			
Methodologies / tests	Ordinary class hours	Student?s personal work hours	Total hours
Guest lecture / keynote speech	21	21	42
Document analysis	0	7	7
Directed discussion	7.5	15	22.5
Laboratory practice	15	30	45
Supervised projects	1.5	15	16.5
Objective test	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 will be dimensioned to be undertaken by groups of students. The size of these groups 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. The use of English in its realization and presentation will be specifically taken into account.
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	<p>The personalized attention to students involves not only the well-known tutoring sessions, but also the following actions:</p> <ul style="list-style-type: none"> <li>- Guidance and monitoring of the work done in the projects/essays/reports and other practices.</li> <li>- Evaluation of the involvement and participation in discussion sessions.</li> </ul>

Assessment		
Methodologies	Description	Qualification
Laboratory practice	<p>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:</p> <ul style="list-style-type: none"> <li>- Assessment of group work, which takes into account the degree of coordination and collaboration among its members.</li> <li>- Personal assessment, which evaluates the specific contribution of one student to the group.</li> </ul> <p>The aspects that will be considered to evaluate these projects are:</p> <ul style="list-style-type: none"> <li>- Accuracy in achieving the objectives using the proposed techniques.</li> <li>- Understanding of the concepts involved.</li> <li>- Originality of the proposals.</li> <li>- Responsibility in delivering the project results in due time, as well as proper use of the established delivery means.</li> </ul>	40
Objective test	Written test divided into two parts: theoretical questions, and modeling of a problem.	40



Supervised projects	The following aspects will be evaluated: <ul style="list-style-type: none"><li>- Knowledge and understanding of presented contents.</li><li>- Knowledge and understanding of the theoretical and practical concepts of the subject involved.</li></ul>	20
---------------------	--	----

#### Assessment comments

Students will need to show balance&nbsp;in their performance on the final examination and the&nbsp;lab practices (group projects). A balance of at least 50% of the corresponding qualification weight will be required on both aspects.

#### Sources of information

<b>Basic</b>	<ul style="list-style-type: none"><li>- Clements, Paul [et al.] (2003). Documenting software architectures : views and beyond. Addison-Wesley</li><li>- Hohpe, Gregor (2004). Enterprise integration patterns designing, building and deploying messaging solutions. Addison-Wesley</li><li>- Sommerville, Ian (2011). Ingeniería de software. Addison Wesley</li><li>- Schmidt, Douglas [et al.] (2000). Pattern-oriented software architecture. John Wiley &amp; Sons</li><li>- Fowler, Martin (2003). Patterns of enterprise application architecture. Addison-Wesley</li><li>- Bass, Len [et al.] (2003). Software architecture in practice. Addison-Wesley</li><li>- Braude, Eric J. (2001). Software engineering an object-oriented perspective. John Wiley &amp; Sons</li></ul>
<b>Complementary</b>	

#### Recommendations

##### Subjects that it is recommended to have taken before

Marcos de Desenvolvemento/614G01052

Validación y Verificación del Software/614G01053

Ferramentas de Desenvolvemento/614G01054

##### Subjects that are recommended to be taken simultaneously

Enxeñaría de Requisitos/614G01027

Aseguramento da Calidade/614G01028

##### Subjects that continue the syllabus

Deseño Software/614G01015

Proceso Software/614G01019

Internet e sistemas distribuidos/614G01023

##### 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.