



Teaching Guide				
Identifying Data			2021/22	
Subject (*)	Software Verification and Validation	Code	614G01225	
Study programme	Grao en Enxeñaría Informática			
Descriptors				
Cycle	Period	Year	Type	Credits
Graduate	2nd 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=614G01053&any_academic=2017_18&			
General description	<p>This subject is intended to master the current solutions in Software Engineering for software validation and verification. These include:</p> <ul style="list-style-type: none">- knowledge on functional and non-functional testing techniques and tools, applicable to different levels (unit, integration, system);- knowledge on techniques and tools for automatic reasoning; and- knowledge on techniques and tools for formal verification.			
Contingency plan	<ol style="list-style-type: none">1. Modifications to the contents2. Methodologies<ul style="list-style-type: none">*Teaching methodologies that are maintained*Teaching methodologies that are modified3. Mechanisms for personalized attention to students4. Modifications in the evaluation<ul style="list-style-type: none">*Evaluation observations:5. Modifications to the bibliography or webgraphy			

Study programme competences / results	
Code	Study programme competences / results
A28	Capacidade de identificar e analizar problemas, e deseñar, desenvolver, implementar, verificar e documentar solucións software sobre a base dun coñecemento adecuado das teorías, modelos e técnicas actuais.
B1	Capacidade de resolución de problemas
B3	Capacidade de análise e síntese
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.
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			
Learning outcomes	Study programme competences / results		
Ability to identify and analyse problems, and design, develop, implement, validate and document software solutions on the basis of a deep and broad knowledge of modern theories, models, and techniques.	A28	B1 B3	C2 C3 C6 C7 C8

Contents	
Topic	Sub-topic
Part I: Software Testing	I.1 Test specification, design, and execution I1.1. Levels and types of tests I1.2. Properties and traceability of requirements I.2 Test management: planning, assessment, metrics and reviews
Part II: Formal methods and automatic reasoning	II.1 Introduction: natural deduction and calculus of sequences II.2 Automatic proof using PVS II.3 What is a theorem prover and what is it used for? II.4 PVS specification language: types, expressions, theories, subtyping II.5 PVS prover: tactics, recursion, equational reasoning
Part III: Model checking	III.1 Introduction to modal temporal logic III.2 Properties specification: deadlocks, safety, liveness, fairness III.3 How a model checker works III.4 Introduction to the use of a model checking tool

Planning				
Methodologies / tests	Competencies / Results	Teaching hours (in-person & virtual)	Student?s personal work hours	Total hours
Guest lecture / keynote speech	B3 C2 C7 C8	21	26.25	47.25
Laboratory practice	A28 B1 B3 C2 C3 C6	14	35	49
Supervised projects	A28 B1 B3 C2 C3 C6	7	7	14
Objective test	B1 B3 C6	3	31.5	34.5
Personalized attention		5.25	0	5.25

(*)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	Master class where the theoretical aspects of the subject are presented.
Laboratory practice	Hands-on student assignment in the lab.
Supervised projects	Student assignments during reduced-group classes.
Objective test	Written test.

Personalized attention	
Methodologies	Description



Guest lecture / keynote speech Laboratory practice Supervised projects Objective test	Questions/answers sessions about theoretical/practical aspects, student assignments, etc. during the office hours of each teacher.
---	--

Assessment			
Methodologies	Competencies / Results	Description	Qualification
Laboratory practice	A28 B1 B3 C2 C3 C6	Hand in and presentation of student assignments, up to a maximum of 4 points in the final score. These are not compulsory to pass.	40
Supervised projects	A28 B1 B3 C2 C3 C6	Student assignments presented during reduced-group classes, up to a maximum of 2 points in the final score. These are not compulsory to pass.	20
Objective test	B1 B3 C6	Written test, up to a maximum of 4 points in the final score. A minimum of 2 points is required to pass.	40

Assessment comments
<p>Those students who do not reach the minimum in the objective test, will be qualified with the qualification they obtain in that objective test.</p> <p>In the second opportunity, the objective test may include a specific evaluation of the laboratory practice.</p> <p>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.</p>

Sources of information	
Basic	<ul style="list-style-type: none"> - Mordechai Ben-Ari (2012). Mathematical Logic for Computer Science. Springer - Ron Patton (2001). Software testing. Sams - Peter Farrell-Vinay (2008). Manage software testing. Auerbach - Kent Beck (2002). Test Driven Development (By Example). Addison-Wesley - Gerard J. Holzmann (2003). The SPIN model checker: primer and reference manual. Addison-Wesley - Mordechai Ben-Ari (2001). Mathematical Logic for Computer Science. Springer - Zohar Manna and Amir Pnueli (1991). The Temporal Logic of Reactive and Concurrent Systems. Specification. Springer - Zohar Manna and Amir Pnueli (1995). The Temporal Logic of Reactive and Concurrent Systems. Safety. Springer
Complementary	

Recommendations
Subjects that it is recommended to have taken before
Software Design/614G01015 Concurrency and Parallelism/614G01018 Software Process/614G01019 Software Architecture/614G01221 Requirements Engineering/614G01222 Quality Assurance/614G01223
Subjects that are recommended to be taken simultaneously
Knowledge Representation and Automatic Reasoning/614G01036 Theoretical Computer Science/614G01039 Development Methodologies/614G01051
Subjects that continue the syllabus
Software Development Projects/614G01226



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.