



## Teaching Guide

Teaching Guide				
Identifying Data			2020/21	
Subject (*)	Software Design	Code	614G01015	
Study programme	Grao en Enxeñaría Informática			
Descriptors				
Cycle	Period	Year	Type	Credits
Graduate	1st four-month period	Second	Obligatory	6
Language	SpanishEnglish			
Teaching method	Hybrid			
Prerequisites				
Department	Ciencias da Computación e Tecnoloxías da InformaciónComputación			
Coordinador	Mosqueira Rey, Eduardo	E-mail	eduardo.mosqueira@udc.es	
Lecturers	, Alonso Ríos, David Monroy Camafreita, Juan Morán Fernández, Laura Mosqueira Rey, Eduardo Pérez Sánchez, Beatriz	E-mail	brais.cancela@udc.es david.alonso@udc.es juan.monroy@udc.es laura.moranf@udc.es eduardo.mosqueira@udc.es beatriz.perezs@udc.es	
Web				
General description	<p>Software Design is a key phase in software life cycle that provides the link between the requirements of a system and its implementation. The most common software design today is based on object-oriented techniques, which consists of developing a program based on objects that interchange messages.</p> <p>This subject will introduce students to the basic elements and properties of object orientation using an object-oriented language like Java. The students will also learn how to represent design artifacts using a modeling language such as the Unified Modeling Language (UML).</p> <p>Finally, the basic principles that represent a good design will be presented and we will learn to identify those typical design problems and their most common solutions represented as design patterns.</p>			



Contingency plan

1. Modifications to the contents

No changes will be made.

2. Methodologies

\*Teaching methodologies that are maintained

Although the teaching methodologies of Guest lecture / keynote speech, Laboratory Practice and Seminar are maintained, the way they are taught changes as detailed in the following section.

\*Teaching methodologies that are modified

- Guest lecture / keynote speech. One group was already being taught telematically. In case of having to teach all of them in a non-presential way what will be done is to record the lecture and use the class hour to solve doubts and perform exercises.

- Laboratory practice. They will be given by telematic means. In the class hours, the teacher will answer the doubts and solve the problems of the students.

- Seminar. The teacher's explanation will be recorded and the class hour will be used to solve students' doubts and problems.

3. Mechanisms for personalized attention to students

- Teams will be used for both synchronous (videoconferencing) and asynchronous (chat) tutorials, respecting the designated tutorial schedules.

- Moodle will be used to communicate important announcements about the subject matter.

4. Modifications in the evaluation

\*Evaluation observations:

The evaluation, in case a face-to-face evaluation can be made, will consist of the following percentages that are in line with the grade report:

- Objective test: 60% (minimum mark of 4)

- Laboratory practices: 40%.

If a face-to-face evaluation cannot be carried out, the weight of the objective test will be reduced following the recommendations of the rectorate and the minimum mark will be eliminated.

- Non-presential objective test: 40% (no minimum score of 4)

- Laboratory practices: 60%.

The non-presential objective test will be a practice (preferably) or a test type if the appropriate conditions do not exist to carry out said practice.

5. Modifications to the bibliography or webgraphy



No changes will be made. All working materials are already available in digital form in Moodle.



Study programme competences / results	
Code	Study programme competences / results
A7	Capacidade para deseñar, desenvolver, seleccionar e avaliar aplicacións e sistemas informáticos que aseguren a súa fiabilidade, seguraza e calidade, conforme a principios éticos e á lexislación e normativa vixente.
A13	Coñecemento, deseño e utilización de forma eficiente dos tipos e estruturas de datos máis adecuados á resolución dun problema.
A14	Capacidade para analizar, deseñar, construír e manter aplicacións de forma robusta, segura e eficiente, elixindo o paradigma e as linguaxes de programación máis adecuados.
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.
C6	Valorar criticamente o coñecemento, a tecnoloxía e a información dispoñible para resolver os problemas cos que deben afrontarse.

Learning outcomes			
Learning outcomes	Study programme competences / results		
	results		
Identify software design as one of the phases of software lifecycle	A7 A13 A14	B3 B4	C3
Know the principles and basic properties of object orientation	A7 A13 A14	B1 B2 B3 B4	C3 C6
Capture software design using the artifacts of a modeling language like UML	A7 A13 A14	B1 B2 B3 B4	C3 C6
Know the basic principles that represent a good software design	A7 A13 A14	B1 B2 B3 B4	C3 C6
Identify typical design problems and their most common solutions	A7 A13 A14	B1 B2 B3 B4	C3 C6
Use a design as a guide for software implementation	A7 A13 A14	B1 B2 B3 B4	C3 C6
Learn an object-oriented language and related aspects (IDE, tests, repositories, etc.)	A13	B1 B2 B3 B4	C3 C6

Contents	
Topic	Sub-topic



1. Introduction	? Software design ? Object-oriented analysis and design
2. Basic Elements of Object Orientation	? Classes and objects ? Object identity ? Object state ? Object behavior
3. Basic Characteristics of Object Orientation	? Abstraction and encapsulation ? Modularity ? Hierarchy ? Polimorphism ? Typing ? Dynamic binding
4. Unified Modeling Language (UML)	? Introduction ? Basic elements of UML ? Static design: Class diagrams ? Dynamic design: Interaction diagrams ? Other diagrams
5. Design Principles	? Quality in design ? SOLID principles ? Types of inheritance
6. Design Patterns	? Introduction to design patterns ? Elementary patterns ? Designs adaptable to changes ? Loosely coupled designs ? Patterns and collections of objects ? Other patterns and principles
Practice	? Introduction to Java ? Exceptions in Java ? Software tests ? Source code repositories

Planning				
Methodologies / tests	Competencies / Results	Teaching hours (in-person & virtual)	Student's personal work hours	Total hours
Guest lecture / keynote speech	A7 A13 A14 B1 B3 C6	30	45	75
Laboratory practice	A7 A13 A14 B1 B2 B3 B4 C3 C6	20	30	50
Seminar	A7 A13 A14 B1 B2 B3 B4 C3 C6	10	10	20
Objective test	A7 A13 A14 B1 B3 C6	3	0	3
Personalized attention		2	0	2

(\*)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 explaining theoretical concepts using different resources: blackboard, projection of digital slides, class notes in electronic format and other resources provided by the teachers in the Virtual Campus of the UDC.



Laboratory practice	Laboratory activities based on the knowledge that students are acquiring in lectures. Students will develop this activities preferably in groups. We will use a modeling tool to build the design artifacts and an object-oriented language (Java) to implement that artifacts.
Seminar	Seminars with activities related to knowledge acquired in lectures or laboratory activities
Objective test	Written test in which the knowledge acquired by students is assessed. Each student must apply their knowledge both in theoretical and practical level.

### Personalized attention

Methodologies	Description
Laboratory practice Seminar	<p>Personalized attention to students includes not only tutorials (either virtual or in-person) to discuss questions, but also the following actions:</p> <ul style="list-style-type: none"> <li>- Monitoring the work of laboratory practices proposed by the teacher.</li> <li>- Evaluation of the results obtained in practice and seminars.</li> <li>- Personalized meetings to answer questions about the contents of the subject.</li> </ul>

### Assessment

Methodologies	Competencies / Results	Description	Qualification
Laboratory practice	A7 A13 A14 B1 B2 B3 B4 C3 C6	<p>Exercises based on Java programming, object-oriented design, testing design, the modeling language UML and the use of design principles and design patterns.</p> <p>Plagiarized practices will be voided, both the original and the copy, and may result in a zero in the overall grade of the given assignment.</p>	40
Seminar	A7 A13 A14 B1 B2 B3 B4 C3 C6	<p>Seminars are practical sessions led by the teacher in which useful aspects related to the assignments are discussed.</p> <p>The seminars do not include the submission of assignments by the students, so it is not an evaluable activity.</p>	0
Objective test	A7 A13 A14 B1 B3 C6	<p>Written test conducted at the end of the semester with theoretical and practical content.</p> <p>It is mandatory to obtain a minimum grade of 4 in the objective test to pass the subject.</p>	60

### Assessment comments

<p>Failure to reach the minimum score in the objective test in any of the opportunities will mean that you can not get more than a 4.5 in the final grade of the subject.</p> <p>Aspects to be considered for the evaluation of second opportunity (July):</p> <p>Laboratory practices grades are the ones obtained at the first opportunity (submission of laboratory practices in the second opportunity is not allowed).Aspects to be considered in the case of part-time enrollment:</p> <p>The obligation to attend activities that require to be in-person is eliminated.</p>
---

### Sources of information



<b>Basic</b>	<ul style="list-style-type: none"><li>- Sierra, K., Bates, B. (2005). Head First Java. O'Reilly</li><li>- Schildt, H. (2018). Java 9. Anaya Multimedia</li><li>- Booch J.; Rumbaugh J. y Jacobson I. (2006). El Lenguaje Unificado de Modelado (2ª ed.) The Unified Modeling Language (2nd ed.). Addison Wesley</li><li>- Martin, R.C. (2004). UML para programadores Java. UML for Java Programmers. Pearson</li><li>- Gamma, E.; Helm, R.; Johnson, R. y Vlissides J. (1996). Design Patterns: Elements of Reusable Object-oriented Software.. Addison Wesley</li></ul>
<b>Complementary</b>	<ul style="list-style-type: none"><li>- Eckel, B. (2007). Piensa en Java (4ª ed.). Thinking in Java (4th ed.). Prentice-Hall</li><li>- Rumbaugh, J.; Jacobson, I. y Booch, J. (2004). The Unified Modeling Language Reference Manual. Addison Wesley</li><li>- Bloch, J. (2017). Effective Java (3rd ed.). Addison Wesley</li><li>- Martin, R.C. (2012). Código limpio : manual de estilo para el desarrollo ágil de software. Anaya Multimedia</li><li>- Freeman, E., Freeman, E., Bates, B. (2004). Head First Design Patterns. O'Reilly</li><li>- Grand M. (2002). Patterns in Java. John Wiley &amp; Sons</li></ul>

## Recommendations

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

Programming I/614G01001

Programming II/614G01006

### Subjects that are recommended to be taken simultaneously

Programming Paradigms/614G01014

### Subjects that continue the syllabus

Software Process/614G01019

Human Machine Interfaces/614G01022

Internet and Distributed Systems/614G01023

### Other comments

It is assumed that students know how to program and understand data structures (Programming II subject) but have never used an object-oriented language. At the beginning of the subject, as the students are introduced to the concepts of object orientation, they will become familiar with the basics of Java programming language.

(\*)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.