UNIVERSIDADE DA CORUÑA

Teaching Guide				
Identifying Data				2017/18
Subject (*)	Software Design		Code	614G01015
Study programme	Grao en Enxeñaría Informática			
Descriptors				
Cycle	Period	Year	Type	Credits
Graduate	1st four-month period	Second	Obligatoria	6
Language	SpanishEnglish			
Teaching method	Face-to-face			
Prerequisites				
Department	Computación			
Coordinador	Mosqueira Rey, Eduardo		E-mail	eduardo.mosqueira@udc.es
Lecturers	Alonso Ríos, David Cabrero Canosa, Mariano Javier Monroy Camafreita, Juan Mosqueira Rey, Eduardo Paz López, Alejandro Pérez Sánchez, Beatriz Sanchez Maroño, Noelia		E-mail	david.alonso@udc.es mariano.cabrero@udc.es juan.monroy@udc.es eduardo.mosqueira@udc.es alejandro.paz.lopez@udc.es beatriz.perezs@udc.es noelia.sanchez@udc.es
Web				
General description	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. 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). 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.			

Study programme competences	
Code	Study programme competences
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.
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 enfrontarse.

Learning outcomes	
Learning outcomes	Study programme competences

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 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 and NetBeans ? Software tests ? Exceptions management ? Use of a source code repository

Planning				
Methodologies / tests	Competencies	Ordinary class hours	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 teacher 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 in groups of no more than two persons. 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	Personalized attention to students includes not only tutorials (either virtual or in-person) to discuss questions, but also the following actions: - Monitoring the work of laboratory practices proposed by the teacher. - Evaluation of the results obtained in practice and seminars. - Personalized meetings to answer questions about the contents of the subject.

Assessment			
Methodologies	Competencies	Description	**Qualification**

Laboratory practice	A7 A13 A14 B1 B2 B3 B4 C3 C6	Two bulletins of exercises based on Java programming, object-oriented design and testing. A design exercise focused on the use of design principles and design patterns. Copied exercises may result in a zero grade, both for the original and for the copy	40
Seminar	A7 A13 A14 B1 B2 B3 B4 C3 C6	Seminars are laboratory practices developed by students with direct assistance of the teacher who, at the end of the laboratory, shows the correct solution of the exercise. Seminars are directly related to theory and practice so the assessment of these activities is delegated to laboratory practice and objective test assessment	0
Objective test	A7 A13 A14 B1 B3 C6	Written test conducted at the end of the semester with theoretical and practical content. It is mandatory to obtain a minimum grade of 4 in the objective test to pass the subject.	60

Assessment comments
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. Aspects to be considered for the evaluation of second opportunity (July): 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: The obligation to attend activities that require to be in-person is eliminated.

Sources of information	
Basic	- Sierra, K., Bates, B. (2005). Head First Java. O?Reilly - Eckel, B. (2007). Piensa en Java (4ª ed.). Thinking in Java (4th ed.). Prentice-Hall - Booch J.; Rumbaugh J. y Jacobson I. (2006). El Lenguaje Unificado de Modelado (2ª ed.) The Unified Modeling Language (2nd ed.). Addison Wesley - Martin, R.C. (2004). UML para programadores Java. UML for Java Programmers. Pearson - Gamma, E.; Helm, R.; Johnson, R. y Vlissides J. (1996). Design Patterns: Elements of Reusable Object-oriented Software.. Addison Wesley
Complementary	- Arnold K., Gosling J. y Holmes D. (2005). The Java Programming Language. Prentice-Hall - Rumbaugh, J.; Jacobson, I. y Booch, J. (2004). The Unified Modeling Language Reference Manual. Addison Wesley - Stevens, P. y Pooley, R. (2006). Using UML. Software Engineering with Objects and Components. Addison Wesley - Freeman, E., Freeman, E., Bates, B. (2004). Head First Design Patterns. O'Reilly - Grand M. (2002). Patterns in Java. John Wiley & Sons

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.