Description of Course Unit

Master in Electrical and Computers Engineering
Programming Laboratories
Instance: 2012/2013
—

Institutional page

General Information

Course Unit: Programming Laboratories
Code: EEC0030
Programmes: MIEEC, 4º, 49 students (on 04.10.2012)
Academic Year: 2012/2013
Semester: 1S
Credits: 6 ECTS
Hours/Weeks: 1x2T, 3x2P
Teacher: João Correia Lopes

Teaching Language

Suitable for English-speaking students

Objectives, Skills and Learning Outcomes

1 - BACKGROUND

Specification, development and maintenance of software applications with GUI using client/server architectures.

2 - SPECIFIC OBJECTIVES

This course aims to:

- Promote the acquisition of software engineering concepts, methods and techniques and enable students to apply them in the design and development of software systems.
- Equip students with practical skills in the use of software development tools appropriate to the specification and development of the product throughout its life-cycle, including debugging, testing and documentation of the Java programming language code.

3 - PREVIOUS KNOWLEDGE

Knowledge of Object Oriented Programming.

4 - PERCENTAGE DISTRIBUTION

Scientific Component: 40%
Technological Component: 60x%

5 - LEARNING OUTCOMES

After completing this course, the student will be able to:

- Identify and document the requirements of a Software System using "user stories"
- Describe the use cases using UML
- Implement a prototype of the User Interface
- Identify and document additional requirements
- Obtain the conceptual domain model using UML
- Obtain business process models using UML
- Describe the architecture using UML
- Validate the architecture through a prototype
- Modelling the structure of classes using UML
- Modelling the behaviour of objects using UML
- Prepare the User manual
- Prepare the Deployment manual
- Write Java classes using standard APIs
- Make versions of software components
- Documenting Java code using Javadoc
- Test the code using Junit
- Use a collaborative documentation development tool
- Use an IDE in software maintenance
- Use a version control system

Program

- Introduction to Software Engineering and Agile Modelling.
- Introduction to Engineering requirements. Requirements documentation.
- UML modelling language.
- Object-oriented software design. Modelling of architecture. Structure, behaviour and architecture design.
- User interfaces design.
- Coding with Java.
- Verification, validation and software testing.
- Software maintenance. Configurations and versioning.
- Project management.

Main Bibliography

- Scott Ambler, *The Object Primer*, Cambridge University Press, 3rd Edition, 2004, ISBN: 978-0-521-54018-6
- Bruce Eckel, *Thinking in Java*, Prentice Hall, 4th Edition, 2006, ISBN: 0131-87248-6 Biblioteca

Complementary Bibliography

- Alberto Manuel Rodrigues da Silva e Carlos Alberto Escaleira Videira, *UML, metodologias e ferramentas CASE*, 2ª Edição, Volume 1, Maio 2005, Centro Atlântico Editora, ISBN: 989-615-009-5 Biblioteca
- Russ Miles e Kim Hamilton, *Learning UML 2.0*, O'Reilly, 2006, ISBN: 978-0-596-00982-3 Biblioteca
- Ian Sommerville, *Software engineering*, Addison-Wesley, 9th Edition, 2010, ISBN: 0-321-31379-8 Biblioteca
- Mauro Nunes e Henrique O.Neill, *Fundamental de UML*, 3ª edição, FCA - Editora Informática, 2004, ISBN: 978-972-722-481-4 Biblioteca

Teaching Procedures

Lectures (2 hours per week) will be used to present the theoretical content, together with practical examples using the methodologies and tools to be used in laboratories. In the laboratory classes (2 hours per week), the students will work in groups of five people in a software project.

Software

- Enterprise Architect (Windows)
- Umbrello (Linux)
- IDE NetBeans (Linux e Windows)
- Dokuwiki
- ArgoUML
- SVN

Keywords

Physical sciences > Computer science > Programming
Physical sciences > Computer science > Programming > Software engineering

Evaluation Type

Distributed evaluation without final exam

Registered evaluation and occupation components

Description	Type	Time (Hours)	Date of conclusion
Attendance (estimated)	Lectures	56	
TP1: User Interface Prototype	Laboratory work or fieldwork	10	2012-10-02
TP2: Requirements Specification	Laboratory work or fieldwork	14	2012-10-16
TP3: High Level Project	Laboratory work or fieldwork	7	2012-11-06
TP4: Prototype	Laboratory work or fieldwork	18	2012-11-13
TP5: Detailed Project	Laboratory work or fieldwork	10	2012-12-04
TP6: Product	Laboratory work or fieldwork	39	2012-12-11
TP7: Product presentation	Attendance	2	2012-12-14
TP8: Individual Performance	Work		
FT1: Use Cases Model	Exercise	1,5	2012-10-08
FT2: Conceptual Domain Model	Exercise	1,5	2012-10-29
FT3: Architectural Model	Exercise	1,5	2012-11-19
FT4: Java	Exercise	1,5	2012-11-26
	Total:	162	

Admission to Exams

Practical work (TP) will be evaluated through the documentation submitted, the application developed and individual performance in the class (TP1 to TP8).

The theoretical concepts are evaluated through the individual response to CAT sheets (FT1 to FT4).

Minimum required to pass this course: 50% in each of the practical components (TP1 to TP8) and 40% overall mark in the CAT sheets (FT1 to FT4).

This course, given its laboratory nature, can not be replaced by taking an exam.

Final grade

Classification = 80% TP + 20% FT

where:

TP = (TP1 + 3* TP2 + 2* TP3 + 2* TP4 + 2* TP5 + 8* TP6 + TP7 + TP8)/20
FT = (FT1 + FT2 + FT3 + FT4) / 4

The classification of any assessment component may vary from element to element in the same group by plus or minus 2 values, based on the opinion of teachers and the self-evaluation to be conducted internally within each group.

Special Assignments

After the demonstration of the product, an oral session may be required for some of the students.

Special evaluation (TE, DA, ...)

Students under special regimes are expected to submit the practical work required for this course as ordinary students.

Students that are not required to be present in the classes, have to present the evolution of their work to the teacher simultaneously with the regular students, and conduct the same theoretical tests.

Improvement of Final/Distributed Classification

Improving the classification requires a new enrollment in the course.

— *JCL*