Universidad de Alcalá

Teaching Guide

Advanced Programming

Bachelor´s degree in

Computer engineering
Computer science

University of Alcalá

Academic year 2019/2020

2nd Year – 2nd Term

GUÍA DOCENTE

Course Name:	**Advanced Programming**
Course Code:	**780014**
Degree programs:	**Computer Engineering** **Computer Science**
Department & subject field:	**Computer Science**
Type:	**Obligatory**
ECTS Credits:	**6**
Year:	**Second Year – Term 2**
Teachers:	José María Gutiérrez, Ana Castillo Martínez
Office hours:	**Dependent on individual tutor**
Classes offered in:	English / Spanish

1. Presentation

This course introduces new concepts and programming techniques that allow students to handle the creation of advanced applications that use concurrency, distribution and Mobile.

To reach these goals, the course will show the origins, evolution, present time and future of the tools, languages and theories behind concurrency.

Entry requirements

Students are required to have previously taken the 'Programming' course.

2. Competences

General skills:

CG4 Ability to define, evaluate and select hardware and software platforms for the development and implementation of systems, services and applications, according to the knowledge acquired as provided in paragraph 5 of resolution BOE-A-2009-12977.

CG6 Ability to design and develop systems or centralized or distributed architectures integrating hardware, software and networks according to the knowledge acquired as provided in paragraph 5 of resolution BOE-A-2009-12977.

Specific skills:

CI8 Ability to analyze, design, build and maintain applications in a robust, secure and efficient way, choosing the paradigm and programming languages most appropriate.

CI11 Knowledge and application of features, functionality and structure of Distributed Systems, Computer Networks and Internet design and implement applications based on them.

CI14 Knowledge and application of fundamental principles and basic techniques of parallel, concurrent, distributed programming and real time.

Outcomes

RA1: Evaluate the effect of programming languages capabilities and limitations on computer systems creation process.

RA2: Describe the evolution of programming languages, paradigms available and their main characteristics.

RA3: Analyze the effect on information systems design of the evolution of programming languages, paradigms available and their main characteristics.

RA4: Apply parallel, concurrent and distributed programming, with the knowledge of their advantages, disadvantages and main algorithm, to the design of more efficient systems.

3. Outline of course contents

Topic 1: Programming paradigms. Programming languages history. Short description of some paradigms. Comparison between paradigms, utility and influence over programming languages.

Topic 2: Introduction to concurrency. Historic evolution and terminology. Concurrency problems. Architectures that allows concurrency. Shared variables. Distributed memory. Justice hypothesis. Safety and vitality properties.

Topic 3: Shared memory concurrency: Monitors. Cooperate and compete processes. Examples. Algorithms, Active waiting, Loks and conditions.

Topic 4: Shared memory concurrency in Java: Semaphores, Critical Regions, Conditional CR and monitors

Topic 5: Java concurrency. JSR166 specification. Concurrency and utilities.

Topic 6: Architectures and patterns for concurrent programming. Tasks and thread pools.

Topic 7: Introduction to distributed programming. Java sockets. Client/Server paradigm.

Topic 8: Distributed systems with Java objects. RMI (Remote Method Invocation). Synchronization and concurrency patterns.

Contents timing

Units	Topics	Total number of hours, classes, credits or time commitment (*)
Programming paradigms and introduction to concurrent programming	1, 2 and 3	2 ECTS
Advanced concurrency and utilities, tasks and thread pools	4, 5 and 6	2,5 ECTS
Distributed concurrency	7 and 8	1,5 ECTS

(*) CATs included - (Continuous Assessment Tests)

4. TEACHING/LEARNING METHODOLOGY. OUTLINE OF ACTIVITIES

Total number of hours: 150 (6 ECTS)

Class contact hours	56 hours + 4 hours of exams
Independent study hours	90 hours
Total hours	150 hours

The course contents previously described shall be taught in the following ways:

- Taught theory classes
- Supervised practical classes: problem solving in class.
- Supervised practical labs.
- Tutorials: individual or group.

In addition, depending on the nature of the work, the students may make use of the following study methods, as well as others:

- Individual realization of coursework but with information input and management as part of a team.

- Exchange of information, problems and doubts which arise during individual work with course mates.

- Organization and production of published journal articles alongside oral presentations and discussions on the results.

- Use of the Virtual Learning Platform as a principal form of access to all activities and subject materials.

Class contact hours:

1. In class: Presentation and discussion of core subject knowledge. Planning and theoretical solving and problems and related hypotheses. Oriented towards the teaching of subject specific skills, especially those related to the key concepts and practices of the imperative programming paradigm.

2. In practical labs: Planning and development of practical exercises which allow problems to be solved and hypotheses to be analyzed, contributing to the development of analytical and critical reasoning skills as well as an understanding of problem solving methods. These will serve as a basis for acquiring the general skills described in part 2 of this guide.

Outside of class:

1. Analyzing and learning course contents, solving of problems, consulting the bibliography, individually preparing coursework, sitting exams and self-evaluation. Oriented especially towards developing personal organization skills and planning work individually or as part of a team.

2. Tutorials: Individuals and group guidance throughout the learning process. Students may attend in person or online.

Materials and resources

- Reference bibliography of core and further reading on the subject.
- Personal computers.
- Development environments and accompanying user guides
- Internet connection.

- Virtual Learning Platform and accompanying user guides.
- Projectors.

5. Assessment

The grading system shall adhere to the '"NORMATIVA REGULADORA DE LOS PROCESOS DE EVALUACIÓN DE LOS APRENDIZAJES" (Regulation of learning assessment procedures) ruled by the Governing Council of the University of Alcalá on the 24 March 2011.

The assessment of students' acquired skills shall take into account the student's attitude and participation in class. Students may choose between Continuous Assessment (PEC: Pruebas de Evaluación Continua) throughout the semester or in certain cases they may request to sit a final exam at the end of the semester if they are able to provide appropriate justification in a timely manner.

Continuous Assessment.

The continuous assessment system consists of: regular submission of laboratory exercises, a final laboratory exercise, a programming exercise in the laboratory and written theoretical/ practical tests which assess taught skills. These may be completed in or submitted to the classroom/ laboratory, or submitted via the Virtual Learning Platform, depending on the case.

- **Grading system for Continuous Assessment.**
  - Each theoretical CAT (T1 or T2) aims to assess knowledge of the material covered in each test. Students will acquire this knowledge by attending classes, studying the course materials prepared by teachers, doing the further reading suggested for each topic, searching for additional material and doing the exercises suggested by teachers or acquired by students.
  - Practical CATs consist of:
    - CATs L1 y L2 involve submitting a piece of practical work dealing with one or several programming problems. Each has a fixed deadline.
    - CAT L3 involves the creation of a completed application which must apply all the knowledge and skills acquired during the course.
    - CAT L4 involves the implementation (in the laboratory or in a designated class) of proposed changes to CAT L3 or to another piece of software or an explanation of the ways in which certain aspects of CAT L3 were implemented in practice.
  - The dates of the CAT examinations shall be given to students in the first days of the course in the 'Course Schedule' which can be found on the Virtual Learning Platform. All deadlines and events taking place during the course can be found here.

- **Weighting of continuous assessments (PEC) in the overall grade.**

The final grade is comprised of 60% theory and 40% practice. The final weighting of each exam/ practical is as follows:

Practical CATs	% of overall grade
L1	2,5
L2	2,5
L3	20
L4	15

Theoretical CATs	% of overall grade
T1	20
T2	25
T3	15

Failing a CAT will not prevent the student from participating in further continuous assessments.

End of term examination.

The end of term examination is only available in certain specified cases and must be requested by students who meet the criteria and who have been granted permission by school administration in accordance with the applicable regulation of the University of Alcalá. Students who take the summative end of term examination will sit an exam which will cover all the theoretical aspects of the course. They must also provide all the practical laboratory work that has been handed in for PEC's L1, L2 and L3, and do a test equivalent to the theoretical PECS and PEC L4.

Examination re-sits.

Examination re-sits shall be held during the month of June for students who failed to pass both continuous assessments and the final exam. It consists of an examination in the same format as the final exam, with the same type of submissions (with a different focus).

Evaluation Criteria:

The following evaluation criteria are established for the course:

CE1. The student knows the history and the characteristics of programming languages and he is able to describe differences between programming paradigms.

CE2. The student understand concurrent programming, its types and evolution from sequential programming.

CE3. The student knows the principles of concurrent programs design.

CE4. The student have acquired basic knowledge about concurrent programs coordination.

CE5. The student have acquired the advanced single memory concurrent application design knowledge.

CE6. The student understand the need of distributed concurrent programming and the differences with single memory concurrency.

CE7. The student knows to program in a language supporting distributed concurrency using sockets or RPC.

CE8. The student knows to develop a single memory concurrent system using any available mechanism of programming languages.

CE9. The student have acquired basic knowledge about advanced concurrent programming.

The next tables show the percentage of marks (0-100) of every test and their relation with evaluation criteria, learning results and competences. The meaning of codes used for evaluation tools is: PEI (plus number) is a theoretical CAT, PL (plus number) is laboratory CAT.

Continuous assessment:

Competences	Learning Outcomes	Evaluation Criteria	Evaluation Tool	Percentage
CG4, CI8, CI14	RA1, RA2, RA3	CE1, CE2, CE3, CE4	PEI1	20
CG4, CG6, CI14	RA1, RA3, RA4	CE5, CE8, CE9	PEI2	25
CG4, CG6, CI11, CI14	RA4	CE6, CE7	PEI3	15
CG6, CI14	RA4	CE3	PL1	2,5
CG6, CI11, CI14	RA4	CE6, CE7	PL2	2,5
CG4, CG6, CI11, CI14	RA1, RA3, RA4	CE5, CE6, CE7, CE8, CE9	PL3	20
CG4, CG6, CI11, CI14	RA1, RA3, RA4	CE7, CE8, CE9	PL4	15

End of term examination and examination re-sits:

In this table the codes are PT for theoretical test, PP for practical assignment and EL for laboratory exam.

Competences	Learning Outcomes	Evaluation Criteria	Evaluation Tool	Percentage
CG4, CG6, CI8, CI11, CI14	RA1, RA2, RA3, RA4	CE1, CE2, CE3, CE4, CE5, CE6, CE7, CE8, CE9	PT	60
CG4, CG6, CI8, CI11, CI14	RA1, RA3, RA4	CE3, CE5, CE6, CE7, CE8, CE9	PP	25
CG4, CG6, CI8, CI11, CI14	RA1, RA3, RA4	CE7, CE8, CE9	EL	15

6. Bibliography

Core reading

- "Java Concurrency in practice". Goetz, Brian / Peierls, Tim / Bloch, Joshua / Bowbeer, Josheph / Holmes, David / Lea, Doug. 2007. Adison Wesley
- "Java in Distributed Systems". Boger, Marko. 2001. Wiley

Further reading

Concurrent programming

- "Concurrent Programming in Java™: Design Principles and Pattern, 2nd Edition". Lea, Doug. 2000. Adison Wesley
- “Principles of Concurrent and Distributed Programming, Second Edition”. M. Ben-Ari. Addison-Wesley. 2006.
- “The Art of Concurrency”. Clay Breshears. O'Reilly Media, Inc. 2009.
- “Concurrency: State Models and Java Programs”. Jeff Magee & Jeff Kramer. John Wiley & Sons. 2006.

Distributed programming

- "Concurrent Systems. Operating Systems, Database and Distributed Systems". Bacon, J. 1998. Addison Wesley
- "Object-oriented Reuse, Concurrency and Distribution". Atkinson, Colin. 1991. Addison Wesley
- "Parallel Program Design". Chandy, Mani / Misra, Jayedev. 1996. Addison Wesley
- "The Java Programming Language". Arnold, Ken / Gosling, James / Holmes, David. 2005. Addison Wesley

- "Concurrent and Real-Time Programming in Java". Wellings, Andrew. 2004. John Wiley & Sons
- "Concurrent Programming Concepts". Brinch-Hansen, P. 1973. ACM Computing Survey
- "Java Distributed Computing". Farley, J. 1998. O'Reilly & Associates, Sebastopol

Programming

- “Exercises in Programming Style”. Cristina Videira Lopes. CRC Press. 2014