Universidad de Alcalá

TEACHING GUIDE

Operating systems

Degree in
Computer Engineering (GIC)
Computer Science Engineering (GII)

Universidad de Alcalá

Academic Year 2023/2024

2nd Year - 1st Semester (GIC+GII)

Approved by the School Board on June 12th, 2023

TEACHING GUIDE

Course Name:	**Operating systems**
Code:	**780007 (GIC+GII)**
Degree in:	**Computer Engineering (GIC)** **Computer Science Engineering (GII)**
Department and area:	**Automática** **Computer Architecture and Technology**
Type:	**Basic (GIC+GII)**
ECTS Credits:	**6.0**
Year and semester:	**2nd Year - 1st Semester (GIC+GII)**
Teachers:	Por definir
Tutoring schedule:	Consultar al comienzo de la asignatura
Language:	English

1. COURSE SUMMARY

This guide is a tool that will allow the student to understand the contents of the course, the competences that will be acquired through its study, the distribution in time of the different activities, and the requirements to pass the subject, as well as other information of interest. It can also be downloaded from the teaching platform available on the Internet for the course.

Operating Systems is a basic subject taught in the first four-month period of the second year of the Degree in Computer Engineering and the Degree in Computer Science. It is the first subject of the Operating Systems discipline, which will be completed during the Degree with Advanced Operating Systems and, finally, with Embedded Systems and Real-Time Systems, the latter two being optional in the Degree in Computer Engineering.

The aim of this course is to introduce the student to the need to use software systems that help to provide sufficiently high levels of abstraction to undertake the development of even more complex systems.

Operating Systems are responsible for making the hardware resources of our platform available to users in a simple and secure way. Their evolution has often been linked to that of Computer Architectures, taking from this discipline a large number of concepts and techniques. In turn, Computer Architectures have evolved to support the requirements that, through Operating Systems, have been imposed by users over time. This mutual feedback is vital for understanding the current state of the discipline, as well as its future trends.

The second topic will consider the different design approaches that can be used to build an Operating System. The student will be introduced to a series of general concepts on the design of large systems specific to this particular discipline. These include layered design, the separation between mechanisms and policies, etc. This will be followed by a study of the interfaces that generally communicate the user and the applications with the Operating System, to conclude with a study of the kernel and the system call mechanisms.

The third topic will allow the student to establish the differences between programs and processes, as well as the structure of both in each of the contexts in which they operate. At the end of this topic, the student will be able to justify the introduction of threads in modern Operating Systems, establish their characteristics, and carry out small programs that make use of them. This topic will conclude with a series of case studies of real Operating Systems. These cases will allow to frame all the theoretical concepts learned previously, as well as particular details specific to each implementation.

The fourth topic presents the different scenarios in which it is necessary for processes and threads to synchronise. Key concepts such as mutual exclusion, atomicity, critical section, etc. will be studied in the context of some of the typical problems. Special emphasis will be put on the mechanisms provided by the Operating System to solve the problems that may arise when multiple processes or threads access shared system resources in an uncontrolled way.

The last topic is dedicated to process and thread scheduling. This topic will introduce the student to the need to select which process should be executed at any given moment in order to improve a series of performance parameters. Also, classical scheduling policies and traffic light synchronisation policies will be studied, ending with the study of the techniques used by some commercial Operating Systems.

Prerequisites and Recommendations

This subject is based on the knowledge acquired by the students in the subjects of Fundamentals of Computer Technology, Computer Structure and Organisation and Fundamentals of Programming, all three of them taught during the two previous four-month periods.

2. SKILLS

Basic, Generic and Cross Curricular Skills.

This course contributes to acquire the following basic, generic and cross curricular skills:

en_CG4 - Ability to define, evaluate and select hardware and software platforms for the development and execution of computer systems, services and applications, in accordance with the knowledge acquired as set out in section 5, annex 2, of resolution BOE-A-2009 -12977.

en_CG6 - Ability to conceive and develop centralized or distributed computer systems or architectures integrating hardware, software and networks in accordance with the knowledge acquired as set out in section 5, annex 2, of resolution BOEA-2009-12977.

en_CG8 - Knowledge of the basic subjects and technologies, which enable them to learn and develop new methods and technologies, as well as those that provide them with great versatility to adapt to new situations.

en_CG9 - Ability to solve problems with initiative, decision making, autonomy and creativity. Ability to know how to communicate and transmit the knowledge, skills and abilities of the profession of Computer Engineering Engineer.

en_CB1 - That students have demonstrated to possess and understand knowledge in an area of study that is based on general secondary education, and is usually found at a level that, although supported by advanced textbooks, also includes some aspects that involve knowledge from the forefront of their field of study.

en_CB2 - That the students know how to apply their knowledge to their work or vocation in a professional manner and possess the competencies that are usually demonstrated through the elaboration and defense of arguments and the resolution of problems within their area of study.

en_CB3 - That students have the ability to gather and interpret relevant data (usually within their area of study) to make judgments that include a reflection on relevant social, scientific or ethical issues.

en_CB4 - That students can transmit information, ideas, problems and solutions to both a specialized and non-specialized public.

en_CB5 - That the students have developed those learning skills necessary to undertake further studies with a high degree of autonomy.

en_TRU1 - Capacity of analysis and synthesis.

en_TRU2 - Oral and written competencies.

en_TRU3 - Ability to manage information.

en_TRU4 - Autonomous learning skills.

en_TRU5 - Team work.

Specific Skills

This course contributes to acquire the following specific skills:

en_CIB4 - Basic knowledge of the use and programming of computers, operating systems, databases and computer programs with application in engineering.

Learning Outcomes

Upon successful completion of this subject/lesson, students will be able to:

RA1. Identify the fundamentals of an operating system, its components and the essential concepts for understanding them.

RA2. Justify the need for operating systems in today's computing environments and their role as an interface between hardware and user programs.

RA3. Distinguish various operating systems and operating environments (traditional, GUI, multimedia, etc.), their differences and resource requirements.

RA4. Install, configure and operate a multi-user operating system.

RA5. Develop applications by using the API of an operating system at the process and thread level.

RA6. Distinguish the most relevant task scheduling techniques for batch, interactive and real-time systems.

RA7. Identify the need for concurrent activities, the problems they cause and the solutions to these problems.

3. CONTENTS

Contents Blocks	Total number of hours

PART 1: Introduction to Operating Systems and background concepts on Computer Architecture.
• Definition of Computer
• Definition of Operating System
• Naked Machine Model
  ◦ Von Neumann architecture.
  ◦ CPU, memory, buses and Input and Output
• Single resident monitor
  ◦ Monoprocessing
  ◦ Monitor protection
  ◦ Monitor service calls
  ◦ Interrupt concept
• Batch processing
  ◦ CPU and I/O usage overlapping
  ◦ I/O architecture and buffering techniques
• Multiprogrammed systems
  ◦ Interrupt-driven I/O
  ◦ Memory protection
  ◦ Dual execution mode: traps and interrupts.
  ◦ Separation of addressing spaces: MMU concept
• Time sharing
  ◦ Protection against CPU monopoly
  ◦ Eviction technique
  ◦ Need for scheduling
• Design techniques:
  ◦ Functional and structural view
  ◦ Difference between mechanisms and policies
  ◦ Hierarchical representation of a computer
	8 hours

PART 2: Operating System Structure
 	
• Restricted and wide views of an Operating System.
• Operating System Functions
• Operating System Interfaces
  ◦ With the user: The shell, Graphical environments
  ◦ With applications: system calls.
• Decomposition in layers of the Operating System
• The kernel of the Operating System
  ◦ Description and basic functions
  ◦ Case studies of designs: Linux, Windows, Mach
• System call mechanisms
  ◦ Description
  ◦ Types of system calls
	12 hours

PART 3: Processes and Threads
 	
• Programs vs. processes
• Structure of a program
• Process concept
• Projection of a program in memory in the user context
• Creating a process
  ◦ Basic life cycle
  ◦ Process-related system calls
  ◦ Kernel context: internal data structures
• Threads
  ◦ Justification
  ◦ Features
  ◦ Processes vs. threads
  ◦ Example of using POSIX threads
	14 hours

PART 4: Synchronization of processes and threads
 	
• Introduction
• Need for communication and synchronization
• Common concurrent scenario types
• Concepts associated with synchronization:
  ◦ Race conditions
  ◦ Atomicity
  ◦ Critical section problem: mutual exclusion, progress and limited waiting.
• Communication and synchronization mechanisms provided by the OS.
  ◦ Pipes
  ▪ Types and operations on pipes
  ▪ System calls for pipes
  ▪ Classic pipe-related synchronization problems
  ◦ Semaphores
  ▪ Operations on semaphores
  ▪ System calls for semaphores
  ▪ Classic synchronization problems with semaphores
  ◦ Mutex and conditional variables
  ▪ Operations on mutex and conditional variables
  ▪ System calls for mutex and conditional variables
  ▪ Classic synchronization problems with mutexes and conditional variables
	12 hours

PART 5: CPU scheduling
• CPU switching mechanism
• Planning concept
  ◦ Rationale
  ◦ Goals
  ◦ Evaluation parameters
• Types of schedulers
• Basic planning policies
  ◦ FCFS
  ◦ SJF
• Priority and requisition concepts
• Advanced planning policies
  ◦ Pure priority planning
  ◦ Round robin
  ◦ Multilevel queuing with and without feedback
• Use cases: Linux and Windows
	10 hours

Week	Contents
1st-2nd	PART 1: Theory (4h) + Practice (4h)
3rd-5th	PART 2: Theory (6h) + Practice (6h)
6th-9th	PART 3: Theory (6h) + Practice (8h) + Evaluation (1h)
9th-12th	PART 4: Theory (6h) + Practice (6h)
12th-15th	PART 5: Theory (6h) + Practice (4h) + Evaluation (1h)
	Written exam (2h)

4. TEACHING - LEARNING METHODOLOGIES. FORMATIVE ACTIVITIES.

4.1. Credits Distribution

Number of on-site hours:	60 hours (56 hours on-site +4 exams hours)
Number of hours of student work:	90
Total hours	150

4.2. Methodological strategies, teaching materials and resources

Face-to-face sessions	• **Theoretical classes:** these classes will be taught in large groups and in them, through lectures, the professor will develop the most important concepts for the understanding of the contents of the subject. • **Resolution of practical cases:** these will be done in small groups. During the sessions, problems that can be solved by means of techniques exposed in class will be presented. The application of these techniques to solve the problem will be guided. • **Presentation of reports and works:** the student will have to present to his classmates and to the teacher reports and projects that he has carried out individually or in small groups. The presentations will make use of the appropriate multimedia techniques. • **Partial tests:** during the development of the course the professor will propose several partial tests to review the acquisition of knowledge and its application.
Personal work	• **Readings** • **Carrying out activities:** exercises, concept maps, exemplifications or information search. • **Participation in forums and activities:** generally, through the teaching platform of the subject.
Tutoring	The tutoring may be both in groups and individually. During the sessions, the professor will be able to evaluate the acquisition of the competencies and will review the reports provided by the students on the work assigned.

Resources

The materials for the preparation of the face-to-face sessions, as well as the activities to be carried out by the student individually, can be found in the Virtual Classroom of the University of Alcalá. The operation of this teaching tool will be detailed in the presentation class of the course; it will be explained, among other aspects, the way in which students will register in the general message forum, usual mechanism of communication with students.

For each activity, the professor will provide a series of bibliographical references that can be consulted in the library of the Polytechnic School.

For those activities that require it, the professor will indicate the way to plan them, as well as the works to be delivered.

5. ASSESSMENT: procedures, evaluation and grading criteria

Preferably, students will be offered a continuous assessment model that has characteristics of formative assessment in a way that serves as feedback in the teaching-learning process.

5.1. PROCEDURES

The evaluation must be inspired by the criteria of continuous evaluation (Learning Assesment Guidelines, LAG, art 3). However, in compliance with the regulations of the University of Alcalá, an alternative process of final evaluation is made available to the student in accordance with the Learning

Assesment Guidelines as indicated in Article 10, students will have a period of fifteen days from the start of the course to request in writing to the Director of the Polytechnic School their intention to take the non-continuous evaluation model adducing the reasons that they deem convenient. The evaluation of the learning process of all students who do not apply for it or are denied it will be done, by default, according to the continuous assessment model. The student has two calls to pass the subject, one ordinary and one extraordinary.

Continous Assessment:

Continuous assessment evaluates the development of competencies throughout the learning process of the subject through a series of summative and formative tests distributed throughout the course. In this way, the student can approach the subject progressively.

This form of evaluation guarantees early feedback in the student's learning process and allows professors, coordinators and other elements of the Quality Assurance System to make a global follow-up, with the possibility of acting in case indicators or specific situations make it advisable.

The evaluation of the part related to the internship will be carried out at the end of the corresponding block.

Assessment through final exam:

This type of evaluation shall be requested in written form in accordance with the regulations of the School Management.

5.2. EVALUATION

EVALUATION CRITERIA

To determine the degree of acquisition of the competences by the student, the skills, attitudes and values demonstrated by the student will be taken into account according to the following evaluation criteria:

CE1: The student has acquired the knowledge about the fundamentals of operating systems and their structure, about processes and threads, as well as about the basic mechanisms of planning and synchronization.

CE2: The student shows ability to apply and integrate the contents to problems, scenarios or case studies related to the subject.

CE3: The student shows ability and initiative to operate the user interface of an operating system.

CE4: The student shows ability and initiative to develop software using the API (Application Programming Interface) of the operating system at the process management level.

CE5: The student shows ability to argue and make judgments on scenarios and case studies presented in the course.

CE6: The student fulfills the assigned tasks.

CE7: The student shows interest in the contents and subject matter.

CE8: The student shows formal care, clarity and rigor in the exposition of ideas and reasoning.

GRADING TOOLS

Continuous assessment system

The evaluation of the students will be carried out progressively during the development of the course. Their performance will be evaluated by their work, knowledge and skills acquired, as well as the improvement of their learning process.

The instruments of continuous evaluation to be used will consist of the realization of activities proposed by the teacher for each of the topics. In global, these activities will suppose 65% of the qualification of the student.

The total of the activities proposed, their type, content and timing, will be communicated to the student during the presentation class and at the end of the corresponding block of content. These activities include:

- PEI: Intermediate Evaluation Tests of test type carried out during the teaching period on the contents of the theoretical parts of the subject. The use of books will not be allowed (25% of the final grade).
- PL, E: Tests carried out on the resolution of practical exercises in the laboratory (40% of the final grade). The evaluation will be carried out considering the development of a set of practices carried out in work teams. The grade will be obtained through individual laboratory tests (PL) of test and/or essay type (short answer, oral defense) and/or through the delivery (E) of such practices. Rubrics will be used as a support tool in the evaluation.

Both types of tests allow to show the student's ability to apply tools for the understanding of the operating system kernel and the operating system interfaces with the user (commands) and with the applications (API), program development tools and/or application of CPU scheduling algorithms.

- PEF: Final Evaluation Test consisting of problem solving at the end of the teaching period (35% of the final grade).

Single exam evaluation system

Those students who have been granted the final evaluation will be evaluated by means of a single global exam on all the contents of the course.

The final evaluation considers the following evaluation instruments:

- PL, E: Individual and group tests on the resolution of practical laboratory exercises (40% of the final grade) carried out during the teaching period (see its description in Continuous Evaluation Systems).
- PEFT: Final Evaluation Test of Test type on the contents of all the theoretical parts of the course (25% of the final grade).
- PEF: Test consisting of problem solving (35% of the final grade).

If the student does not pass the course in the ordinary call, in continuous or single evaluation, he/she will have the possibility of taking an extraordinary exam with the same evaluation instruments described for the final evaluation.

The student will find in the virtual course of the subject all the details about the evaluation tests: dates, contents, formats, etc. Likewise, it is very important that you take into account Article 34.3 of the Learning Assessment Regulations, regarding the originality of papers and tests:

"Plagiarism is understood as the copying of texts without citing their source and giving them as their own elaboration and will automatically lead to the grade of failure (0) in the work or tests in which it had been detected. The professor who notices signs of plagiarism in the evaluation works or tests presented to him/her shall report this fact to the person responsible for the degree within a maximum period of two days, so that he/she may proceed, if necessary, to inform the Rector in case it could constitute a disciplinary infraction or a crime.

GRADING CRITERIA

This section shows the relationship between learning outcomes, criteria, instruments and grading. In the following tables PLi refers to the laboratory test associated to practice i, and Ei to the deliverable of the corresponding practice i.

Ordinary call, Continuous assessment system

Skills	Learning outcomes	Evaluation criteria	Grading tool	Grading weight
CG8, CB1-CB5, TRU1-TRU5, CIB4	RA1-RA3, RA6, RA7	CE1	PEI1, PEI2	25%
CG4, CG6, CG8, CG9, CB1-CB5, TRU1-TRU5, CIB4	RA3-RA5	CE2, CE3	PL1	15%
		CE2, CE4-CE8	E2+PL2, E3	15%, 10%
CG8, CG9, CB1-CB5, TRU1-TRU5, CIB4	RA1-RA7	CE1-CE8	PEF1	35%

Ordinary call, final evaluation, and extraordinary call

Skills	Learning outcomes	Grading criteria	Grading tool	Grading weight
CG8, CB1-CB5, TRU1-TRU5, CIB4	RA1-RA3, RA6, RA7	CE1	PEFT1	25%
CG4, CG6, CG8, CG9, CB1-CB5, TRU1-TRU5, CIB4	RA3-RA5	CE2, CE3	PL1	15%
		CE2, CE4-CE8	E2+PL2, E3	15%, 10%
CG8, CG9, CB1-CB5, TRU1-TRU5, CIB4	RA1-RA7	CE1-CE8	PEF1	35%

In order to pass the course, it is a requirement to pass the laboratory practices completed during the course, since they constitute the practical part, according to article 6.4 of the Learning Assessment Regulations.

6. BIBLIOGRAPHY

6.1. Basic Bibliography

- Sistemas Operativos. S. Sánchez Prieto. Segunda edición. Servicio de Publicaciones de la Universidad de Alcalá. 2005

6.2. Additional Bibliography

- **Theory**
  - Sistemas Operativos Modernos. A. S. Tanenbaum. 3ª edición. Prentice Hall, 2009.
  - Operating Systems. Internals and Design Principles. W. Stallings. 8ª edición. Prentice Hall, 2009.
  - Operating Systems Design and Implementation. A. S. Tanenbaum y A. S. Woodhull. 3th

edition. Pearson Prentice Hall, 2006.
- Operating System Concepts. A. Silberschatz, P. B. Galvin y G. Gagne. 9th edition. Wiley, 2015.
- Unix. Programación avanzada. Francisco M. Márquez García. Tercera edición. Ra-Ma, 2004.
- Computer organization and architecture. William Stallings. 10th edition. Pearson Prentice Hall, 2015.

Practice

- Unix y Linux, guía práctica. S. Sánchez Prieto y O. García Población. 3ª edición. Ra-Ma, 2004.
- El entorno de programación UNIX. B. W. Kernighan y R. Pike. 1ª edición. Prentice Hall Hispanoamericana S. A., 1989.
- Unix, programación avanzada. F. Márquez García. 3ª edición. Ra-Ma, 2004.
- El lenguaje de programación C. B. W. Kernighan y D. Ritchie. 2ª edición. Prentice Hall Hispanoamericana S. A., 1991.
- C/C++. Curso de programación. F. J. Ceballos Sierra. 5ª edición. Ra-Ma, 2019.

Disclosure Note

During the evaluation tests, the guidelines set out in the Regulations establishing the Rules of Coexistence of the University of Alcalá must be followed, as well as the possible implications of the irregularities committed during said tests, including the consequences for committing academic fraud according to the Regulation of Disciplinary Regime of the Students of the University of Alcalá.