Universidad de Alcalá

TEACHING GUIDE

OPERATING SYSTEMS

Bachelor's Degree in
Computer Engineering
Computer Science

Universidad de Alcalá

Academic year 2019/2020

2nd Year – 1st Term

Subject name:	Operating Systems
Code:	780007
Degree:	Degree in Computer Engineering Degree in Computer Science
Department and area:	Computer Engineering dpt. *(Dpto. Automática)* Computer Architecture and Technology area
Type:	Basic
ECTS credits:	6
Course and term:	2nd year, 1st term
Teaching staff:	Óscar García Población
Tutorship hours:	To be determined. See up-to-date information in UAH's Blackboard
Language:	English

1. Presentation

This guide specifies the contents of the subject, the competences that will be acquired by studying it, the timing of the different activities and the requisites to pass the subject as well as other data of interest. It will be available for download in the educational web platform for the subject in https://uah.blackboard.com

Operating Systems is a basic subject scheduled in the second term of the first course of the Degree on Computer Engineering and Degree in Computer Science. It is the first subject of the matter of Operating Systems, which will continue with Advanced Operating Systems and, optionally, with Embedded Systems and Real Time Systems.

The goal of this subject is to introduce the student to the need of using software systems that help providing abstraction levels high enough to accomplish the development of even more complex systems. Operating systems are in charge of making hardware resources available to the users in a simple and safe manner. Their evolution has often been linked to that of Computer Architectures, being conditioned by this discipline in a large number of concepts and techniques. In turn, Computer Architectures evolved to support the requisites imposed by the users through Operating Systems. This mutual feedback is vital to explain the current state of this discipline, as well as to understand its future trends.

The first theme will focus on studying this interrelation, emphasizing those elements of Computer Architecture that are required to approach the study of Operating Systems.

The second theme studies the different design approaches that can be employed to build an Operating System. The student will be introduced to a series of general concepts related to the design of large systems customized to this specific discipline. Worth to be mentioned are: the layered design, the separation of mechanisms from policies, etc. Next, the interfaces that usually communicate users

and applications with the Operating System will be studied. Finally, the kernel and the system call mechanisms will be studied.

The third theme will allow the student to establish the differences between programs and processes, as well as their structure in the context in which they operate. At the end of the theme, the student will be able to justify the introduction of threads in the modern Operating Systems, establish their characteristics, and write small programs that make use of them. This theme concludes with a series of case studies of real operating systems. These case studies provide a frame for all the theoretical concepts learned previously, as well as particular details of each implementation.

The last chapter is dedicated to the CPU scheduling and inter-process synchronization. With this chapter, the student will understand the need to carry out a selection of which process should run next at every moment, in order to improve some performance parameters. The classic scheduling policies and the techniques used by several commercial Operating Systems will be studied.

1.1. Prerequisites and Recommendations

This subject build on the knowledge acquired by students in the subjects Computer Technology Concepts (*Fundamentos de Tecnología de Computadores*), Computer Structure and Organization (Estructura y Organización de Computadores) and Programming Fundamentals (*Fundamentos de Programación*), both taught in the previous term. It is therefore highly recommended to have passed these subjects before approaching the study of Operating Systems.

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.

CG8 Knowledge of basic materials and technologies that enable learning and development of new methods and technologies, as well as to equip them with great versatility to adapt to new situations.

CG9 Ability to solve problems with initiative, decision making, autonomy and creativity. Ability to communicate and transmit the knowledge and skills of the profession of Technical Engineer.

Specific skills:

CIB4 Basic knowledge of the use and programming of computers, operating systems, databases and software with applications in engineering.

Learning outcomes

RA1: Identify the foundations of an Operating System, its components and the basic concepts.

RA2: Justify the need of the Operating Systems in nowadays computing environments and their role as an interface between the hardware and the user software.

RA3: Differentiate types of Operating Systems and operation environments (traditional, graphical user interface (GUI), multimedia, etc.), their characteristics and requirements in terms of resource needs.

RA4: Install, configure and operate a multiuser operating system.

RA5: Develop applications by means of the Application Program Interface (API) of the Operating System, focusing in processes and threads management.

RA6: To Choose the adequate scheduling policy depending on the type of the operating system being batch, interactive or real-time.

RA7: To Identify the need to perform tasks concurrently, being aware of the problems that this implies, and the possible solutions to them.

3. Contents

Contents program

Part	Themes	Total hours
PART 1: Introduction to Operating Systems and previous concepts of Computer Architecture	- Definition of Computer - Definition de Operating System - Bare machine model • Von Neumann architecture. • CPU, memory, buses and input/output (I/O) - Resident monitor • Single processing • Monitor protection • Calls to monitor services • Concept of interruption - Batch processing • CPU and I/O usage alternation • I/O architecture and buffering techniques. - Multiprogram systems • Interrupt-based I/O • Memory protection • Dual execution mode: traps and interrupts. • Separation off address spaces: concept of MMU - Time sharing • Protection against CPU appropriation • Pre-emption technique • Need for scheduling - Design techniques: • Functional and structural view • Difference between mechanisms and policies • Hierarchical representation of a computer	12h

PART 2:
Operating System structure
	- Restricted and wide views of an Operating System
	- Functions of the Operating System
	- Operating System Interfaces
	- Operating System interfaces
	  • With the user:
	  ▪ The shell
	  ▪ Graphic environments
	  • With applications: system calls
	- Decomposition of the Operating System in layers
	- The Operating System kernel
	  • Description and basic functions
	  • Design case studies: Linux, Windows, Mach
	- System call mechanisms
	  • Description
	  • System call types
		12h

PART 3:
Processes and threads
	- Programs vs. processes
	- Program structure
	- Concept of process
	- Memory mapping of a program in user context
	- Process creation
	  • Basic life cycle
	  • Process creation related system calls
	  • Kernel context: internal data structures
	- Threads
	  • Justification
	  • Characteristics
	  • Processes vs. threads
	  • POSIX threads usage example
	  • Threads and processes synchronization
	  • Semaphores
	  • Synchronization with semaphores
	  • Classic synchronization problems that can be solved with semaphores
	- Case studies: Linux and Windows
		18h

PART 4: CPU usage scheduling	- CPU context switch mechanism - Concept of scheduling • Justification • Goals • Evaluation parameters - Scheduler types - Basic scheduling policies • FCFS • SJF - Concepts of priority and preemption - Advanced scheduling policies • Pure priority scheduling • Round Robin • Multilevel queue scheduling with and without feedback - Case studies: Linux and Windows	14h

Schedule

Week	Contents
1st - 3rd	PART 1: Theory (6h) + Practice (6h)
4th - 6th	PART 2: Theory (6h) + Practice (6h)
7th - 10th	PART 3: Theory (8h) + Practice (10h)
11th - 15th	PART 4: Theory (6h) + Practice (8h)
	Final exam (2h)

4. Teaching-Learning methodologies. Formative activities.

4.1. Credits distribution

Number of presence hours:	56 hours + 4 hours of assessment
Number of hours of student's independent work:	90 hours
Total hours:	150 hours

4.1. Strategies methodologies, materials and didactic resources

Presence classes	• **Theory classes**: master classes, imparted in large groups. The lecturer will develop the main concepts for the comprehension of the contents of the subject. • **Resolution of practical cases**: carried out in reduced groups. These sessions will present different problems to be solved with the techniques exposed in class. In a guided manner, students will apply the said techniques to solve the problems. • **Presentation of reports and works**: the student will present, to his/her classmates and the lecturer, reports and projects carried out either individually or in small groups. The presentations will make use of the appropriate multimedia techniques. • **Partial tests**: During the course the instructor will propose several partial tests to check the acquisition and application of knowledge.
Autonomous work	• **Readings** • **Activities**: exercises, conceptual maps, examples, search for information. • **Participation in forums and activities**, usually through the teaching platform of the course.
Tutorship hours	Tutorships might be carried out in groups or individually. in them, the lecturer will be able to evaluate the acquisition of competences and revise the reports provided by the students regarding the assigned works.

Materials and resources

The materials for presence sessions, as well as the materials for activities to be carried out individually by students, will be published in BlackBoard, https://uah.blackboard.com. The use of this tool will be detailed in the presentation class of the subject. The lecturer will explain, among other things, the way in which the students must inscribe themselves in the general message's forum, which will be an important mechanism of communication with the students.

For every activity, the lecturer will provide a series of bibliographic references that can be accessed in the library of the School.

For those activities that require it, the lecturer will specify the schedule as well as the deliverables that should result from its completion.

5. EVALUATION: Procedures, evaluation and grading criteria

5.1. Procedures

Evaluation can be done following either continuous evaluation or with a final exam. For each alternative there are two examination sessions: the ordinary and the extraordinary one.

Continuous evaluation

The continuous evaluation considers the development of the competencies during all the learning procedure (NRPEA, Art. 3) by using a set of additive tests distributed the semester, so the student can learn the subject progressively. This guarantees the early feedback in the learning procedure of the student, and allows the teachers, coordinators and stakeholders in the Quality Assurance System to perform inspections and to react to certain indicators or situations.

The evaluation of the practical contents will be done at the end of its corresponding block.

Extraordinary evaluation.

It will take place the day and hour stated in the exam calendar made publicity available by the *Junta de Escuela* of each centre.

5.2. Evaluation criteria:

In order to determine whether the student has achieved the planned goals or not, the next evaluation criteria will be taken into account:

CE1: The student has acquired the knowledge about the fundamentals of the Operating Systems and their structure, processes, process scheduling and process synchronization.
CE2: The student shows the ability to apply and integrate the concepts learnt to problems, scenarios or study cases related with the subject.
CE3: The student has the skills and initiative to operate the user interface of an Operating System.
CE4: The student has the skills and initiative to develop software using the Operating System API that makes use of processes management.
CE5: The student has the skills to justify and diagnose typical use case scenarios of the matter.
CE6: The student fulfills the required tasks.
CE7: The student shows interest for the subject.
CE8: The student is methodical, clear and formal when reasoning about the ideas or concepts related with the subject.

5.3. Grading instruments

Continuous evaluation process

The grading process will be continuous during the duration of the course. The students' performance will be evaluated according to their work, knowledge, acquired skills, and enhancement of their learning process. There will be **continuous evaluation activities** proposed by the lecturer for every theme. In general, these activities will constitute a 65% of the grade. Every activity will be proposed during the presentation class or at the end of the corresponding contents block, along with their scheduling and contents. These activities include:

- **IET**: **I**ntermediate **E**valuation **T**ests. Tests, taken during the teaching period, about theory contents. The use of documentation will not be allowed in these tests. (25% of the final grade)

- **LT-P**: **L**aboratory **T**ests and **P**ractical assignments, done in groups or individually, about two main subjects (40% of the final grade):

  - User interface (**LT**). Individual **L**aboratory **T**ests, multiple choice and/or short questions. They will reveal the student's skill dealing the console of an Operating System. (20% of the final grade).

  - Application Program Interface. **P**. Problems to be resolved in groups or individually about software development using the Operating System API. The submitted code will be evaluated with a grading rubric. They will reveal the student's ability to use development tools to write software that uses the processes and threads services offered by a modern Operating System (20% of the final grade).

- **FEE: F**inal **E**valuation **E**xam with two practical problems to be done at the end of the teaching period (35% of the final grade).

Single exam evaluation process

The students that, having been granted for the final exam in the ordinary evaluation, and that fulfill the specific requirements detailed below for the practical exercises will be evaluated in a single exam of all the subject's contents. This **single exam evaluation** will be composed of the following evaluation instruments:

1. **LT-P**: **L**aboratory **T**ests and **P**ractical assignments, done individually, about the main subjects of the course (40% of the final grade).
2. **SE1**: Multiple choice exam about the theory contents of the course (25% of the final grade)
3. **SE2**: Problem solving exam with practical application of the theory contents (35% of the final grade)

Students who do not pass this exam will be able to sit an extraordinary exam with the same format and conditions of the ordinary exam described above.

The students will find in the web page all details about testing activities such as dates, contents, formats, etc.

The regulations of the university (*Normativa de Evaluación de los Aprendizajes, aprobada en Consejo de Gobierno 2016*) states that:

Plagiarism, defined as the copying of text without citing sources and presenting them as his/her own work, automatically leads the student to fail in the subject in which it is detected. This result should be without prejudice to the disciplinary responsibilities in which students may incur.

5.4. Grading criteria:

In this section we will establish the relationship between learning outcomes, evaluation criteria, evaluation instruments and final grading.

Ordinary Continuous evaluation

Skills	Learning outcomes	Evaluation criteria	Evaluation instruments	Grading weight
CG8, CIB4	RA1 - RA3, RA6, RA7	CE1	IET1, IET2	25%
CG8, CG4, CG9, CIB4	RA3-RA4	CE2, CE3	LT1, P1	20%
CG8, CG6, CG9, CIB4	RA3-RA5	CE2, CE4-CE8	TL2, P2	20%
CG8, CG9, CIB4	RA1-RA7	CE1-CE8	FEE	35%

Extraordinary evaluation and single exam evaluation

Skills	Learning outcomes	Evaluation criteria	Evaluation instruments	Grading weight
CG8, CIB4	RA1-RA3, RA6, RA7	CE1	SE1	25%
CG8, CG4, CG9, CIB4	RA3-RA4	CE2, CE3	TL1, P1	20%
CG8, CG6, CG9, CIB4	RA3-RA5	CE2, CE4-CE8	TL2, P2	20%
CG8, CG9, CIB4	RA1-RA7	CE1-CE8	SE2	35%

It is a necessary condition to qualify for the course to pass the practices, because they are considered to be the practical part of the subject (Normativa de Evaluación de los Aprendizajes)

As a general rule, the students that do not participate in all the practical activities will be graded as not present.

6. Bibliography

Basic Bibliography

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

Complementary Bibliography

Unix y Linux, guía práctica.
Sebastián Sánchez Prieto, Óscar García Población
Ra-Ma, 3 Ed., 2004
Operating Systems.
Stallings, William.
Unix. Programación avanzada.
Francisco M. Márquez García. Tercera edición.
Ra-Ma 2004.
Operating Systems Concepts.
A. Silberschatz, P. B. Galvin and G. Gagne.
Modern Operating Systems.
A.S. Tanenbaum.
Operating Systems: Design and Implementation.
Tanenbaum, A.S. and Woodhull, A.S.
Computer Organization and Architecture.
William Stallings.

Computer architecture: a quantitative approach.
Hennessy, John L., Patterson, David A., Cuarta edición.
Elsevier 2006.