# Course Syllabi

**CODE** 209

**NAME** Operating Systems

**Credits** 6 ECTS

**Period** Spring Semester

**Course Specifications**

The course includes lectures about fundamental concepts involved in the internal design of any Operating System. It also includes laboratory work about programming complex software services on the POSIX of the UNIX family of operating systems.

It requires deep and strong knowledge of the hardware structure of the computer (knowledge of previous subjects like Computer Technology and Computer Structure is compulsory, as the concepts will appear frequently during description of the implementation of the OS).

***Note for Erasmus students***

If you come from a **non Computer-Science or Information Technology degree**, you are advised to **not choosing this subject**.

If your Erasmus supervisor recommends you to take it, please refer him or her to contact the lecturers (guille@ac.uma.es) to discuss the suitability of this subject. We have a recurring series of badly advised students that fail to understand the theory of the subject because of the lack of previous knowledge.

Just being an experienced programmer is not enough as a base to approach this subject.

**Objectives**

The subject focuses on the Operating System as an interface between the developer and the computer hardware in a general-purpose system. Any kind of usage of the computer is done through the Operating Systems and never directly:

- Many abstractions offered by programming languages (like threads, processes, files, network sockets, memory allocation, etc.) are not part of the programming language. Instead, the OS implements them and the languages just call the through system calls. This is why all programming languages offer the same abstractions.
- Security is also implemented in the Operating System. The concept of user and access permissions does not exists in the hardware but created and managed by the code of the OS. That includes controlling the access to the hardware resources by the programmer.

With these concepts in mind, you will understand that High Level Languages are only one more layer on top of OS software. You will become a better programmer understanding that any language is just generating code that uses the common OS

implementation of resources. In this sense, you will be able to focus on management of resources more than on the language itself.

*Contents*

1. Processes and threads.
2. Process scheduling.
3. Memory management.
4. Input/output subsystem.

Laboratories

5. The UNIX programming environment: Programming on the POSIX standard.
6. Low-level binary files.
7. Communication through the Operating System (shared memory and queues).
8. Signals and processes.
9. Graceful termination of service programs.

**Assessment**

For the standard student, continuous evaluation of the progress is compulsory. There will be NO FINAL exam at the end of the course.

If the student misses an intermediate exam laboratory work (with exception of properly documented causes like medical reasons, severe illness or decease of close relatives, etc.) the final score will be NOT EVALUATED.

Assessment involves two parts. To pass the subject, it is required to pass both parts with a minimum score of 5:

- Evaluation of knowledge of theory and solving of problems during the course. This evaluation will be done through two kinds of events:
    - Intermediate exams.
    - Exercises interleaved with theory classes. Attendance to classes is compulsory.
- Evaluation of the laboratory work. Evaluation will include two kind of events:
    - Handing in the output of the laboratories through the Virtual Campus.
    - Exercises interleaved with laboratory sessions. Attendance to sessions is compulsory.

# Course Syllabi

**Lecturers**

Dr. Ricardo Quislant: (theory sessions)
quislant@uma.es
Room 2.2.25

Dr. Guillermo Pérez (laboratory sessions)
gperez@uma.es
Room 2.2.34