Systems Architecture

Academic Year: (2019 / 2020) **Review date: 25-04-2019**

Department assigned to the subject: Telematic Engineering Department

Coordinating teacher: IBAÑEZ ESPIGA, MARIA BLANCA

Type: Compulsory ECTS Credits : 6.0

Year : 2 Semester : 1

REQUIREMENTS (SUBJECTS THAT ARE ASSUMED TO BE KNOWN)

Programming and Systems Programming

OBJECTIVES

1. The student must be able to design a software system using the C Programming Language containing non-trivial data structures, dynamic memory management, and using engineering techniques to translate a set of given high level constraints, derived from a hypothetical industrial setting, into a robust application.

2. The student must be able to use proficiently the following industry-category tools: a compiler with different options to generate debugging information and to analyze the diagnostics produced while developing the application, an Integrated Development Environment (IDE) to implement a software system, a version controlled system to handle regular development flows, a cross compiler to create multidevice versions of an application and conduct experiments to verify device compatibility, and profiling tools to analyze memory behavior in a software application.

3. The student must be able to: work effectively in a team to execute a project entailing the design of a software application on a mobile device, generate ideas collaboratively in a team to promote the exchange of information, organize the work in a team to optimize its performance and comply with the project requirements, and divide tasks effectively among the team members.

4. The student must be able to: learn autonomously, manage different information sources, generate and value concise information about the tasks accomplished, manage the time of personal work, and present effectively the results derived from the process.
he exchange of information, organize the work in a team to optimize its performance and comply with the project requirements, and divide tasks effectively among the team members.

DESCRIPTION OF CONTENTS: PROGRAMME

The programme is divided into the following blocks:
1. The C programming language
1.1. Basic data types and flow constructions
1.2. Structure of a C application. The pre-processor, division in files and creating an executable.
1.3. Pointer manipulation.

2. Dynamic memory management in C
2.1. Dynamic data structures
2.2. Memory leaks
2.3. Concurrent tools
2.4. Tools for detecting memory leaks

3. Architecture of the Linux
3.1. Kernel, processes, and filesystem
3.2. Main libraries
3.3. Concurrency

4. Team project design
4.1. Conflicts and their resolution
4.2. Project development

LEARNING ACTIVITIES AND METHODOLOGY

The activities used to underpin the competences and the skills in the course are (preceeded by the reference to the program objectives):

- Exercises covering the following topics: design the most appropriate data structure for a functionality in a mobile application, write code fragments to manipulate data structures, read/write fields, process data, etc, calculate the amount of memory occupied by different data structures (PO: a).

- During the lab sessions code fragments are written, compiled, linked and executed using different compiler options and detect, analyze and correct these programs using the debugger (PO: b).

- During the lab sessions code fragments are written to create, destroy and manipulate data structures using dynamic memory. Students are also requested to divide a given functionality into functions and write their code (PO: c).

- During an eight-week period students are divided into teams of four or five members and they must execute a project entailing the design of a software application containing multiple milestones, deliverables and objectives(PO: d).

- Write detailed meeting minutes with the action items and final conclusions, exchange information between teammates using chats, forums, email, explain the requirements derived from the specification of a work module, and the solution decided by the team (PO: g).

- Students are requested in several activites throughout the course to search for auxiliary documents to support the information studied in a topic. In their final report, they must acknowledge the information sources they used (PO: i).

- Use of the following tools: Virtual machines, compiler, IDE, version control and emulator in multiple laboratory sessions (PO: k).

During these activities the teaching staff reviews the student work in the class, supervises the lab sessions, answers questions in course forum, maintains at least one hour a week of office hours and calls for plenary office hours upon demand.

ASSESSMENT SYSTEM

Continuous evaluation (EC) for the course (60%):
- Individual lab (10%)
- Partial exam (10 %)
- Team project development (30%)
- Project test (10%)

Final Exam (EF)(40%)
- There is a minimum of 40% for this test

There are 10% of additional points for unscheduled activities.
These points will be directly added on EC.

% end-of-term-examination:	40
% of continuous assessment (assigments, laboratory, practicals…):	60

BASIC BIBLIOGRAPHY

- Steve Oualline: Practical C Programming, Proquest, 1991