SOFTWARE APPLICATION DEVELOPMENT CYCLE FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS CASE STUDIES

I. Goals

- To understand the way in which functional and non-functional requirements for a software application can be stated.
- To describe the case studies used during the practical applications classes.

II. Theory Overview

See Lectures (Chapter I)

A project's most important steps are known as phases. The sequence of all stages from start to finish is known as the project's life cycle.

For developing a software application the following models can be used:

- Cascade – with the following phases: Requirements, Design, Implementation and Testing in this order.
- Prototype – in the first stages a prototype is being developed in which various alternatives are experimented. Afterwards the real application is developed.
- Incremental – the application is shipped in increments, adding features on each increment.
- Spiral – the application is developed starting from a base skeleton on which components can be developed/ added.

Details about this development models are offered in the Lectures (Chapter I – Subchapter called "Models for software applications development"), alongside instructions regarding the opportunity of using them in an appropriate context.

Regardless the chosen model of development, for the proper development process it is important that requirements regarding the final project's result be clearly specified, and offering a complete and accurate description of what must be developed in the project. These requirements must be understood by all people working on the project (regardless of their skills), that's why documents must not include useless technical specifications or details, but must depict what the project must accomplish in an accessible language.

Documents must be ranked from general to specific, for allowing a simpler tracking of the specifications, until the desired level of detail is achieved according to the person's role in the project(manager, client, developer, etc.).

A working plan which can be used in the case of IT application development (software/ hardware) can be:

1. General description of the application's functionality (starting from the main actors and their allowed permissions, actions);
2. Detailing functional requirements in the Use Cases document – this document will describe the main and alternative workflows for each use-case, the initial/final state, imposed requirements, possible extensions;
3. Detailing non-functional requirements in the Supplementary Specifications document;
4. Terms Dictionary – this dictionary may eliminate confusions from presentation and may help shorten presentations.

Section III pictures an example on how to fill in those documents. Certainly it is not a widely accepted standard. There are other options – which enhance clarity and simplicity during reading, a more complete description can be considered.

Of course, these requirements can be updated (modified, added upon to) while the project is on going. For example, some specifications can be detailed after the design has been approved. Furthermore for the prototype development model some requirements can be modified, even be dropped.

The resulting documents should highlight the history of all changes as well, if they occur.

III. Examples

Case 1: Client – Server application for signing up for classes and recording-visualizing grades.

D1. Terms Dictionary

Lecture	Course provided by the university
Lecture offer	Course alternative (the offers if an active course in different days with different professors)
Lectures catalog	A list with all course offers
Faculty	All the professors
Class	All the students signed up for a lecture offer
Schedule	The selected choices of all the students and professors during a semester
Grade	The result of a student evaluation for a lecture
Grades report	All the grades obtained by a student during a semester
Grades card	The grades obtained by a student in all semesters

D2. Problem Statement

The application will replace the existing software and assure data migration from the old database.
Required functionality:

Students:

- May register to lecture offers.

- In the lectures catalog they will find information about professors, imposed teaching line, knowledge base, credits, schedule, etc.
- In the first registration period, students can choose more options than necessary (options will be processed in the order they are selected), the application will not accept schedule overlaps or selecting lectures related to failed imposed courses.
    - Students will be accepted in a course offer based on their grades from previous semesters, in ascending order
    - A course offer is validated if the minimum number of students register for it
    - A maximum number of students per course offer is also imposed
- At the end of the first registration period students are notified if they have the enough number of credits and which of their selections have been validated. They'll also be notified of the second registration period and the remaining offers.
- Students can view the schedule and their grades.

Professors: Can join a course offer, can view student groups, and can view/modify grades.

Secretary: Upgrades the information about students, professors, courses, registration periods, evaluations, etc.

D3. Supplementary requirements (not included in use cases):

Functionality	Multi – user
Compatibility	Windows interfaces
Reliability, usage times	24h, 7 days/week, pause periods < 10%
Performance	2000 simultaneous users, 1 visualization < 2 sec
Security	Only the student can change his course offers, only the professor can change grades, etc.
Restrictions	Data transfer from the old database

D4. Use Cases – general description

- Login;
- Professor's information;
- Student's information;
- Logout;
- Lecture selection;
- Class evaluation;
- Lecture offers selection;
- Grades visualization, etc.

4.1. Login

Description	The users are allowed or not to access the application.
Base flow	The actor fills in the username and password fields.

	The system validates the password and allows the actor to access the application.
Alternative flows	If the username or the password are not correct The system will show an error message. The actor will choose between going back to the base flow or leaving the application.
Initial state	Login screen
Final state	Success – the actor is logged in and the application's menu is listed Failure – unchanged state, login screen prompted or exit with error message.
Special requirements	Username and password codification
Possible extensions	Not available

2nd Case: Non-linear process identification program (Ex. Thermic system – Oven).

3rd Case: Designing a PID regulator software for controlling a water recirculation system.

4th Case: Black and white image processing and object detection program.

5th Case: Program for stock management of merchandise in a factory's department store and the semi-automatic control of stock.

IV. Working plan

1. For the third application, two more use cases will be described.
2. For one of the second, third, fourth or fifth application it will be presented the description using the forms proposed in III.1.
3. Write a few sentences on the chosen application's development in a prototype and incremental model (see lectures).