Universidad
Carlos III de Madrid
www.uc3m.es

Final Thesis

Title: *E-LEDA: E-Learning Data Analyser*

Author: *Sohrab Farzaneh Candón*

Degree: *Computer Engineering*

Thesis Director: *Dr. José Arturo Mora Soto*

Date: *30/04/2013*

Universidad Carlos III de Madrid
www.uc3m.es

State of the Document

State of the Document	
Document Title	Final Thesis
Project	E-Learning Data Analyser (E-LEDA)
Author	Sohrab Farzaneh Candón
Company	Universidad Carlos III de Madrid (UC3M)

VERSION HISTORY					
Version	**Author**	**Description**			**Date**
		Included	*Modified*	*Deleted*	
1.0	S.F.C.	Initial Version			15/10/2012
1.1	J.A.M.S.	- State of the art Review	-	-	23/11/2012
2.0	S.F.C.	-	- Formats - Template	-	22/04/2013
2.1	S.F.C.	- Working Methods In E-LEDA project - Traceability Matrixes	- Reference update - Technology Versions Update	-	23/04/2013
2.2	S.F.C.	- Prioritisation - Planning Integration in the document	-	-	24/04/2013
2.3	S.F.C.	- Budget	- Sections Adjustment	-	25/04/2013
3.0	S.F.C.	- Conclusions	-	-	30/04/2013

Content Table

Figure Table

Tables Table

Abstract

The *E-LEDA: E-Learning Data Analyser* project describes the complete detailed development of the *E-LEDA system*, which includes an *iOS app,* a *Web app* and *Cloud Services* for providing a lecturer a monitoring tool to analyse the student performance when the courses are complemented with an e-learning platform evaluation. This document describes the complete analysis of the *E-LEDA* project and the design, implementation and test of the first basic *iOS app*. The main objective of the *E-LEDA project* is to provide the higher quality of the final product by the usage of a combination of tools such as PSP, the Pomodoro Technique, UML and other techniques. The project follows the incremental-iterative use case driven lifecycle described by the Larman's method.

The complete development of the *E-LEDA project* is estimated to be 12 months and one week, counting with two junior software engineers and one senior engineer at part time for the project, The cost of the system is *52.867,90€ - fifty two thousand eight hundred and sixty seven euros and nighty cents,* VAT included.

As an *E-LEDA project* aside product a working method have been developed for this project as a combination of PSP, the pomodoro techniques, and Larman method, resulting in an improvement of the planning estimation accuracy for this project by an 18,1%.

Resumen

El proyecto *E-LEDA: E-Learning Data Analyser* detalla el desarrollo completo del sistema *E-LEDA*, que incluye una aplicación para *iOS,* una aplicación *Web* y una serie de *Servicios en la nube* para permitir a un profesor monitorizar el desempeño de sus alumnos cuando el curso de una asignatura se complementa con el soporte de una plataforma de e-learning para el desarrollo de las actividades de evaluación. Este documento describe el análisis completo del proyecto *E-LEDA,* el diseño detallado, la implementación y los test de la primera fase de la construcción del proyecto, una aplicación básica para *iOS*. El objetivo principal del proyecto *E-LEDA* es el de proporcionar un producto final con la máxima calidad a través del uso de una combinación de técnicas como PSP, la técnica Pomodoro, UML y otras técnicas. El proyecto sigue el modelo iterativo-incremental guiado por casos de uso descrito en el método de Larman.

El desarrollo completo del proyecto *E-LEDA* está estimado en 12 meses y una semana contando el mismo con dos ingenieros de software junior y un ingeniero senior a tiempo parcial para el proyecto. El coste del sistema es de *52.867,90€ - cincuenta y dos mil ochocientos sesenta y siete euros y noventa céntimos* IVA incluido.

Como producto adyacente del proyecto *E-LEDA*, se ha desarrollado un método de trabajo para la realización de este proyecto a través de la combinación de PSP, la técnica pomodoro y el método de Larman, resultando en una mejora del 18,1% en la precisión de la estimaciones para la planificación en este proyecto.

Agradecimientos

Este proyecto culmina un largo recorrido de 5 años y no podría haber sido finalizado sin el apoyo de mucha gente que ha estado a mi lado en distintas circunstancias y momentos. Este proyecto no se habría podido realizar sin el apoyo de mi madre, Nieves, que siempre me ha mantenido con los pies en el suelo, ni el de mi padre, Nasser, que siempre me ha ayudado a pensar a lo grande, a definir mis objetivos y me ha ido empujando poco a poco para llegar hasta donde estoy ahora. Este proyecto no habría sido posible tampoco sin la paciencia de mi hermano, Sergio, ni de sus votos de silencio forzosos en mis muchas horas de trabajo en casa, ni de Jorge, mi segundo padre, que me ha criado durante prácticamente toda mi infancia.

Este proyecto está especialmente dedicado a tres personas que me han aportado tres visiones diferentes del mundo desde mi infancia, mi abuela Nieves, con quien he compartido momentos buenos y menos buenos, mi abuelo Miguel, que me enseño a descubrir el mundo desde pequeñito, y mi abuelo Valentín, sin el cual hoy no estaría donde estoy puesto que me enseñó a abordar los problemas y proyectos desde una forma diferente (desde jugar al ajedrez, a construir un monopatín).

Sin duda este proyecto no habría sido abordable sin la inestimable ayuda de mi tutor y amigo Arturo, quien me guió con los planteamientos del proyecto y a abordarlo desde un punto de vista diferente aportando increíbles ideas. También la ayuda de mis compañeros de laboratorio Marce, Yulls, Cynthya y Manuel ha sido muy importante en el desarrollo de este proyecto, desde la discusión de distintas ideas hasta los cafés con tarta y las comilonas, y Consuelo, mi profesora de informática cuando era aún un niño, quien me enseño a pensar de forma diferente para abordar los problemas que pudieran surgir desde que tenía 9 años.

Gracias a mis compañeros por la inestimable ayuda que me han ofrecido durante estos años, en especial a Dani, que me ha salvado y ayudado incontables veces durante estos años, al trolletariado al completo, porque las risas son una parte fundamental en la vida de todo ingeniero, a Alberto y Jorge, por enseñarme que media hora de trabajo puede llegar a cundir mucho, incluso si las dos horas anteriores se han dedicado a buscar por los rincones más recónditos de internet, y a Omar por criticarme todos mis diseños con su mirada artística y por todas las horas de bailoteo nocturno desde que mi mente alcanza a recordar.

Gracias al consejo de sabios, Mario, Martín y Cristian, por las largas horas de tetería, festivales, conciertos, viajes y conversaciones interminables sobre cualquier tema (desde chicas a la economía mundial), gracias por esos inestimables consejos desde tres puntos de vista tan distintos.

Mi desarrollo como persona no habría sido posible sin la ayuda de toda mi familia de cerca y de lejos (Madrid (España), Málaga (España), Ibiza (España), Londres (UK), Barnsley (UK), Teherán (Irán), Texas (US), etc…) que siempre me ha apoyado y me han enseñado como afrontar los problemas.

Por último, pero no menos importante, a todos los amigos que he conocido por mis aventuras por el mundo, que me han enseñado cosas que no se pueden aprender en ningún libro y con quienes he descubierto que la única frontera es la que tú mismo impongas y que de un ambiente multicultural y multidisciplinar se pueden extraer conocimientos y experiencias increíbles.

Gracias a todos, sin vuestra ayuda nunca habría llegado hasta aquí. El próximo destino, el mundo.

Sohrab Farzaneh Candón

State of the Art

1 Introduction

The Aim of the *E-LEDA: E-Learning Data Analyser* project is to develop an *iOS* application able to process and present data from an E-Learning platform such as *Chamilo* or *Moodle* in order to help the lecturers to perform data analysis about the state of their courses.

The objective is to create a tool able to show charts and results of the students performance in a specific subject or group of subjects so the lecturer can analyse the suitability of the E-Learning platform installed and the evolution of the students. The main idea is to keep the application as simple as possible so the information is presented in the most direct way to the user.

The *E-LEDA project* side objective is to provide a high quality, highly extensible and maintainable product by using a combination of software engineering tools and methodologies, providing a detailed design for the system.

One of the client's requirements is to develop the application using *iOS*. *iOS* is also one of the most extended mobile operating systems in the business field and it requires a specific hardware to work with (*iPhone* 3GS or later, *iPod* 4th generation or later, *iPad* support is out of the scope of this project for *iOS 6.1*).

The *E-LEDA project* is divided in different documents in order to provide the information in the most dynamic and maintainable way possible, this allows to view or modify independently different parts of the *E-LEDA project*. By the time of writing this document the *E-LEDA project* is composed by four different documents:

E-LEDA Final Thesis: Describes the state of the art and briefly describes the different parts of the *E-LEDA system* development. It also includes the conclusions and the description of the methods and methodologies used while developing the *E-LEDA system*, serves as a merge document for the results on the rest of the documents (This document).

E-LEDA Software Requirement Specification: Includes the specification of the complete *E-LEDA system* and the requirements of the system in detail describing what the system shall do (Farzaneh Candón, SRS, 2013).

E-LEDA Use Case Specification: Includes the complete definition of the analysis uses cases of the *E-LEDA system,* describing how the system shall behave in detail (Farzaneh Candón, UCS, 2013).

E-LEDA Construction Phase 1: Defines the complete behaviour of the *E-LEDA system* for the first construction phase including planning, detailed use cases, design, implementation and tests (Farzaneh Candón, CP1, 2013).

In the future, the different construction phases shall be also allocated in independent document in order to maintain the atomicity of the structure followed in the *E-LEDA project.*

2 iOS Architecture and Versions

iPhone OS was presented in the *Macworld conference & Expo* in 2007 in San Francisco by *Steve Jobs* as an operating system for the first *iPhone,* including a user interface based on a *multi-touch* screen (Apple Inc., 2007).In 2010 the *iPhone 4* was presented in the *Apple WWDC* (Apple Inc., 2010) and the name of the operating system was changed to *iOS.*

2.1 iOS Architecture

iOS is a mobile operating system based in OS X and Darwin BSD. *iOS* has a layered architecture composed by four layers with different levels of abstraction. Figure 2.1-1 iOS Layers represents the name and some of the functions of each layer forming the *iOS* architecture (Apple Inc., 2012).

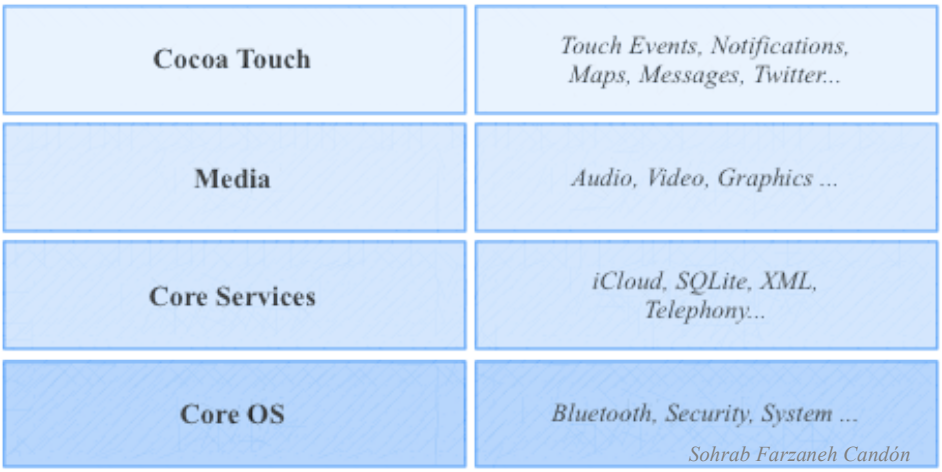

Figure 2.1-1 iOS Layers

Cocoa Touch Layer groups the *frameworks* and functionality to provide high lever services such as touch events, storyboarding (since *iOS* 5), multitasking (since *iOS 4*), peer-to-peer services (since *iOS 3*), etc. It also provides a collection of *frameworks* that allows the developer to handle advertisements, messages or maps.

Media Layer allows the programmer to use specific graphic, audio and video technologies, it also provides a series of *frameworks* in order to work with the integrated core video, audio or graphical technologies. It provides also *iOS* specific OpenGL *frameworks* for 2D and 3D content.

Core Services Layer includes system services such as *SQLite*, *iCloud*, XML or Automatic Reference Counting. This system services increase the development efficiency allowing faster programming by providing high-level features for common problems. It also includes *frameworks* for network connections or data management.

Core OS Layer contains the low-level system services like threading, networking or math computation, it also includes *framework* for managing low-level features such as Bluetooth, security or external accessories compatibility.

2.2 iOS Versions

The first version of the *iPhone OS*[1] was presented in 2007 (Apple Inc., 2007) since the first release until the last one presented in *Apple Special Event* on 12th September 2012 six major versions and over forty revisions have been released. Table 2.2-1 iOS Versions represents the different major releases and the last versions of each of them. It also presents the discontinued versions for older devices[2].

Version	Release Date	Last Revision	Last Revision Date	Devices *(Minimum required)*	Status
1.x	June, 2007	1.1.5	July, 2008	*iPhone (1st G.)* *iPod Touch (1st G.)*	Obsolete
2.x	July, 2008	2.2.1	January, 2009	*iPhone (1st G.)* *iPod Touch (1st G.)*	Obsolete
3.x	June, 2009	3.1.3	February, 2010	*iPhone (1st G.) –limited* *iPod Touch(1st G.) –limited* *iPhone 3G*	Discontinued *Final release 1st G*
		3.2.2	August, 2010	*iPad*	Obsolete
4.x	June, 2010	4.1	September, 2010	*iPhone 3G –limited* *iPod Touch (2nd G.) –limited* *iPhone 3GS*	Obsolete
		4.2.1	November, 2010	*iPhone 3G –limited* *iPod (2nd G.) –limited*	Discontinued *Final release 2nd G*
		4.3.5	July, 2011	*iPhone 3GS* *iPod Touch (3rd G.)* *iPad*	Obsolete
5.x	June, 2011	5.1.1	May, 2012	*iPhone 3GS* *iPod Touch 3rd G* *iPad*	Discontinued *Final release 3nd G*
6.x	September, 2012	6.1.3	March, 2013	*iPhone 3GS –limited* *iPod (4th G.) –limited* *iPad 2 –limited* *iPhone 4S* *iPod (5th G.)* *iPad (3rd G.)*	Current

Table 2.2-1 iOS Versions

As shown in Table 2.2-1 iOS Versions some devices do not support any version updating since a certain release, from now own those devices will be called *obsolete devices* although some of them might be currently in use. Figure 2.2-1 iOS Version Timeline represents a timeline showing the last versions available for the *obsolete devices* and the evolution of the different *iOS* versions.

[1] Later known as *iOS* (see *section 2* iOS Architecture and Versions)
[2] Data Source: iOS a Visual History (Vox Media Inc., 2013)

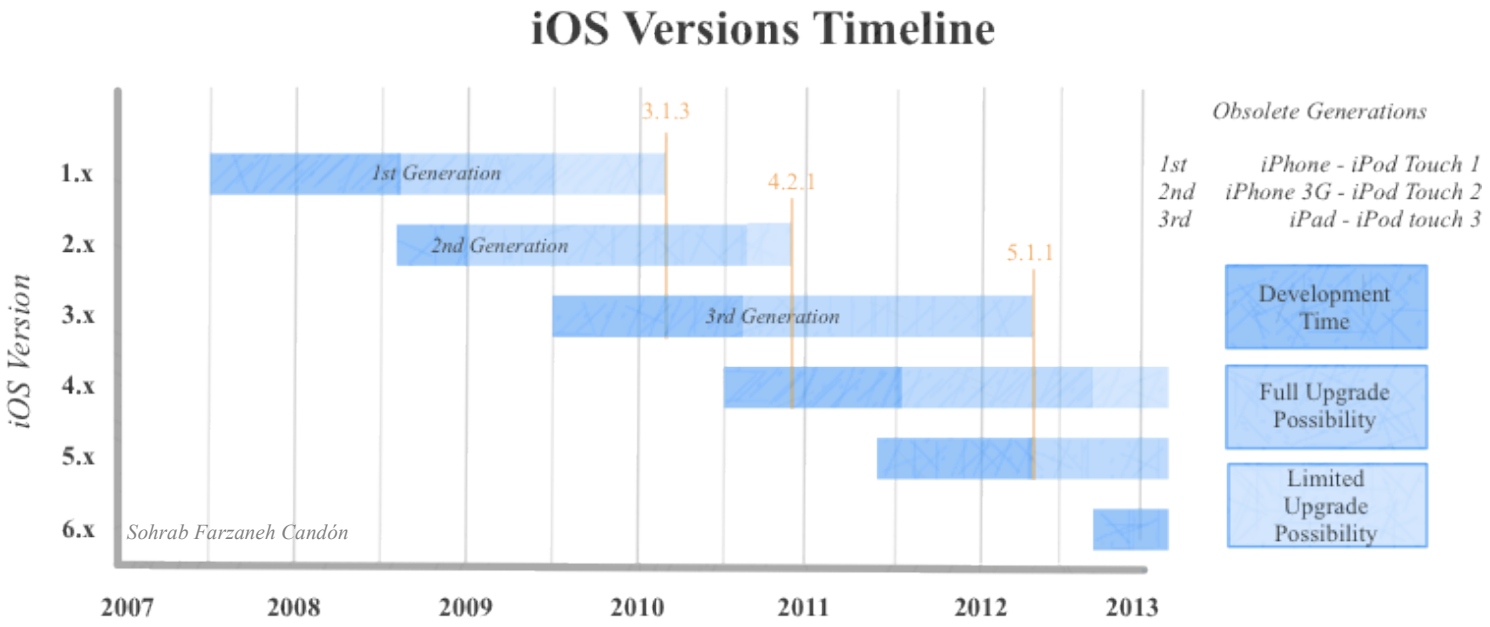

Figure 2.2-1 iOS Version Timeline

It is important to remark that some updates are allowed but limited for some older devices, Figure 2.2-1 iOS Version Timeline represents also upgrading possibility and the limited upgrading possibility.

3 iOS Development Tools

iOS 6.1.3 will be the operation system version used for the development of the *E-LEDA system*, because it is the last version of the operating system and it is compatible with all Apple's mobile devices released since middle 2010 (*see section 2.2 above*). In order to develop iOS application some different tools shall to be used. The required programming language to develop an iOS application is *Objective-C* and it´s most common *IDE* is *XCode*, in order to improve the results and minimize the development time third-party frameworks will be used in addition to the iOS frameworks. This section will explain briefly what *Objective-C* is, which advantages provide *XCode* and the third-party framework used in the development of the *E-LEDA system*.

3.1 Objective-C

Objective-C is an object oriented programming language developed in the 1980's by StepOne. Objective-C derives and uses some of the syntax of *Smalltalk* and it is a strict *C* superset. Objective-C can be compiled with GCC and it includes also some other features such as dynamic typing or the usage of categories that allow changing the functionality of a program without having the original source code.

Objective-C does not include any support for the user interface; therefor the *model-view-controller* design pattern is encouraged while programming in Objective-C.

The *model-view-controller* design pattern allows to present information, store data and manage the application logic in different modules, so any of the modules can be managed in an independent manner. The Figure 3.1-1 Model-View-Controller shows the structure and the communication between the different modules of the architecture (Microsoft, 2012).

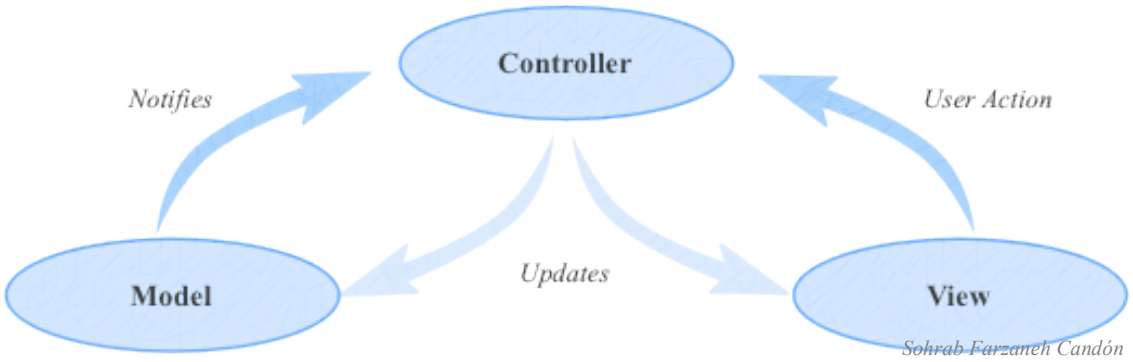

Figure 3.1-1 Model-View-Controller

The Figure 3.1-1 Model-View-Controller represents an adaptation to the original model, which receives the user actions directly in the controller. In the case of iOS application the user perform actions directly in the screen, which also represent the view; that is why the *model-view-controller* pattern had to be adapted. (Krasner & Pope, 1988) (López Hernández, 2008) (McGalliard & Agapow, 1997).

3.2 XCode 4

XCode 4 is the last mayor version (by the time of writing this document) of the *Integrated Development Environment (IDE)* that Apple provides to program *iOS* applications. *XCode* includes *LLVM* and *GCC* compilers, and it is able to compile C/C++, Objective-C, Objective-C++, Java and AppleScript, it also provides refactoring tools, version control integration (*Git or Subversion*), debugging tools, interface builder and the *iOS SDK*, among many others. As external tools it also includes iOS Simulator, and instruments for performance and behaviour analysis (Apple Inc., 2012). Using *XCode* 4.5 no code can be released for *iOS* 4.3 or older.

3.3 Frameworks

In words of Dick Riehle in his dissertation in the *Swiss Federal Institute of Technology* (Zürich):

> *A framework is a class model, together with a free role type set, a built-on class set, and an extension-point class set.*

(Riehle, 2000)

Another way to express the words of Riehle is saying that a framework is a collection of functions, classes *(in object-oriented programming)* and other resources helping software development. *iOS* SDK offers a large amount of frameworks that helps the developer to program applications. This section does not include information about the iOS frameworks (see section 2 above, for more information), but information about third-party frameworks used in the *E-LEDA system*.

3.3.1 Charts

In order to simplify the chart construction a specific framework will be used. The Table 3.3-1 iOS Chart Frameworks shows a comparison of different frameworks taking into account some relevant aspects for the project.

Framework	Multi-Touch	Multiple Charts	Documentation	Licence	Last Stable Version	Comments
CorePlot (Core-Plot, [s.d.])	YES	YES	Wiki, Web	BSD	1.2 *(Nov, 2012)*	Wide documentation
iVisualization (iVisualization, 2011)	NO	NO	Not Found	Private	1.0 *(Jan, 2011)*	iOS 4.2 (Obsolete)
PowerPlot (NuAS, 2013)	YES	YES	Web	GPLv3	1.4.3 *(Nov, 2012)*	On-line resources
S7GraphView (s7graphview, 2012)	YES*	NO	Not Found	BSD	*??* *(Aug, 2012)*	Many contributors, with different versions
Tapku (Ross, 2013)	YES	NO	Document	Apache	?? *(April, 2013)*	Includes also Coverflow, calendar and grids.

Table 3.3-1 iOS Chart Frameworks[3]

From the data in Table 3.3-1 iOS Chart Frameworks, it turns clear that the most complete frameworks for the project purpose are *CorePlot, PowerPlot,* and *Tapku*, because the *E-LEDA system* requires a framework with pie, bar and line chart capabilities able to handle multi-touch events.

Tapku framework include much more functionality than required, this fact could decrease the application time efficiency and could harden the application development.

PowerPlot includes the required functionality for the *E-LEDA system;* it also includes user interaction features.

CorePlot also includes the required functionality for the *E-LEDA system,* among other charting capabilities with a wide documentation.

CorePlot is the chosen option, because it is the simplest option including all the needed capabilities and has a wide documentation.

4 Methodologies and Methods

This section describes the methods and methodologies used for the *E-LEDA project*, and how each of them is going to help in the development process.

4.1 Larman's Method

The *Larman's Method* is an agile[4] development method based on the *rational unified process (RUP)* designed by *Craig Larman* (Larman, 2002). The aim of this method is to simplify the complexity of RUP and provide to un-experienced developers the possibility of using a simple agile methodology.

The *Larman's Method* is an incremental, iterative use case driven process, divided in three big modules:

1. **Planning and Requirement Specification:** The first module of the *Larman's Method* consists in planning, requirement specification, use cases construction for the entire system and providing a wide overview of the complete system.
2. **Construction:** The second part of the *Larman's Method* is divided in several sub-modules depending on the project. Those sub-modules are called construction phases. The program is divided in different construction phases taking into account the use case specification from the *planning and requirement specification* phase *(use-case driven)* then, each construction phase is developed on top of the previous one *(incremental)*. One construction phase goes from analysis to implementation and test; at the end of each construction phase one part of the system is fully functional *(iterative)*.

[3] Data Source: iOS Frameworks (Allwein, 2012). Actualized on April 23rd 2013
[4] Some people might discuss whether this is an agile methodology or not.

3. **Installation:** The last module of the *Larman's Method* is the installation of the entire system. This is the only deliverable in the project, with the full functional system.

The Figure 4.1-1 Larman's Method Lifecycle represents the *Larman's Method* lifecycle

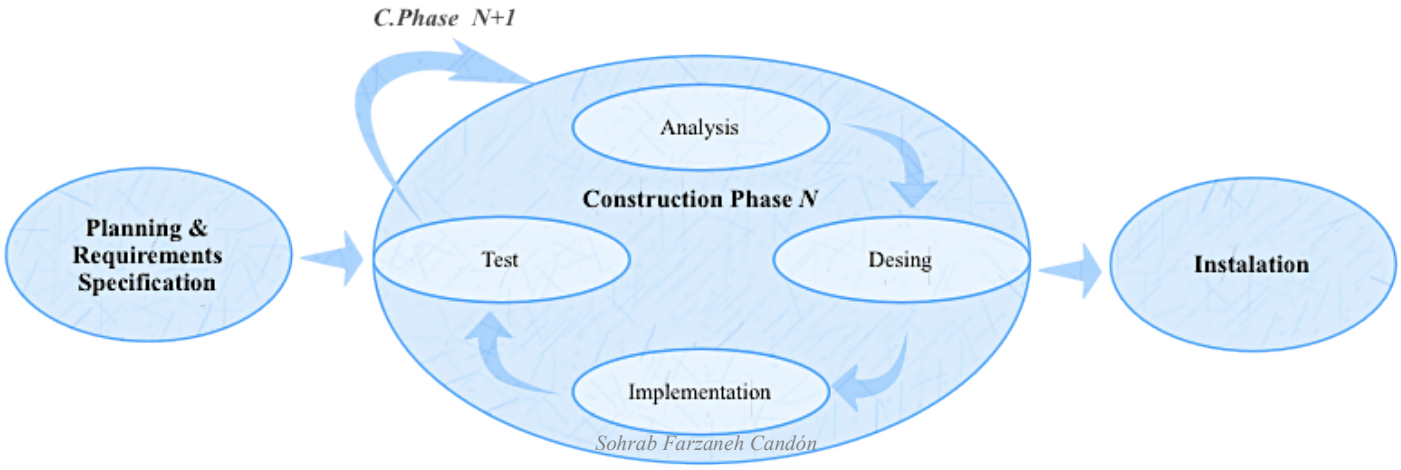

Figure 4.1-1 Larman's Method Lifecycle

4.2 Personal Software Process (PSP)

The *Personal Software Process (PSP)* is a structured software development process created by *Watts Humphrey* (Humphrey, 2000). PSP's objective is to increase the efficiency and productivity of an engineer by measuring and analysing the needed time for performing a certain task such as designing, compiling or programming. *PSP* emphasize the product quality and prioritize the reduction of introduced error by reviewing the performed task instead testing them.

PSP is divided in three main phases each of them divided in two sections *(X.0 & X.1)*.

- **PSP 0:** introduces the process discipline and measurement
- **PSP 1:** Introduces estimating and planning
- **PSP 2:** Introduces quality management and design

For each of the *PSP* phases the developer shall write one or more programs. After mastering the *PSP* process the developer is able to work in a *Team Software Process (TSP)* working team. The Figure 4.2-1 PSP Levels shows the different *PSP* levels in detail and the associated task to each of them.

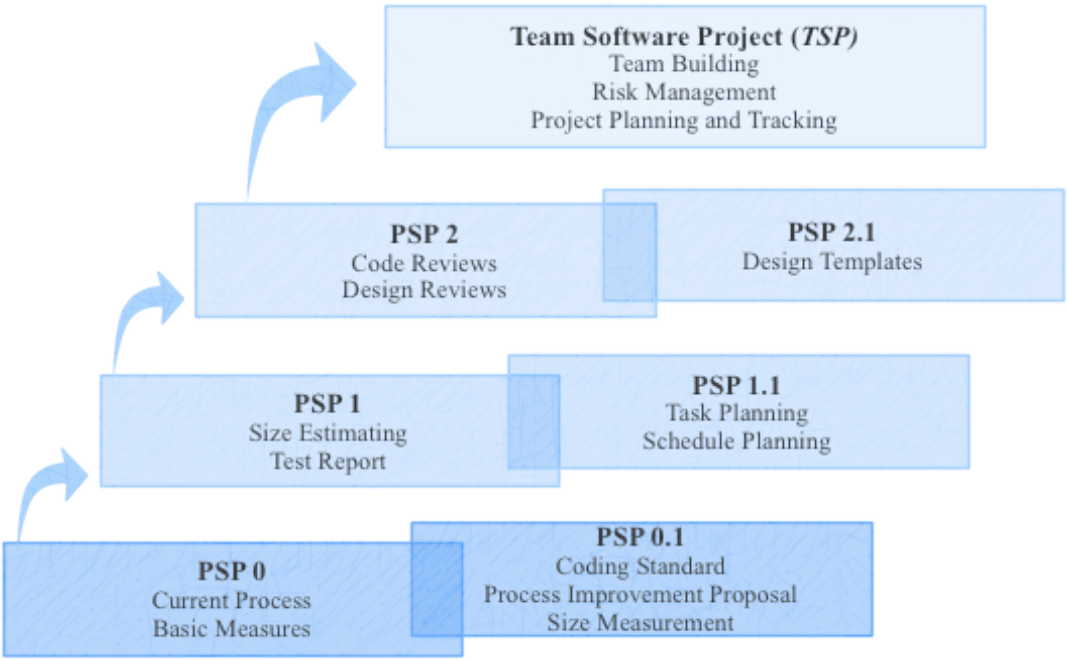

Figure 4.2-1 PSP Levels[5]

One of the problems of the PSP is the complexity of its tasks and measures and therefor the PSP learning curve, for the development of the *E-LEDA system* a simplification of PSP will be used. The aim of those modifications is to simplify the complexity of the PSP processes and combine it with the pomodoro technique (explained in section 4.3 Pomodoro Technique).

4.3 The Pomodoro Technique®

The *Pomodoro Technique*[6] is a time management technique developed by *Francesco Cirillo* consisting in the division of the working activities in fixed time intervals with a short break between each interval (Cirillo, 2009). This technique helps to organize the work and keep the focus in the objective while doing a certain task.

The *Pomodoro Technique* recommends a working time of 25 minutes and a 3-5 minutes break between each working period (or *pomodoro*), after four *pomodoros,* a longer break of 15-30 minutes is recommended. The technique describes five main objectives needed to obtain the maximum benefits of the usage of the *pomodoro technique:*

1. Find how much effort an activity requires
2. Cut down on interruptions
3. Estimate the effort for activities
4. Make the *pomodoros* more effective, by reviewing the work in each of them
5. Set up a time table

The pomodoro technique will be used for the project development as a time unit, helping to estimate the effort needed for each task, and for the working time monitoring.

[5] Data Source: Introduction to TSP(Over, 2010)
[6] The Pomodoro Technique® and Pomodoro™ are registered trademarks of Francesco Cirillo.

4.4 Working Method in E-LEDA project

In order to improve the quality of the *E-LEDA system* and the efficiency while working on every aspect of the project a merge of the different methods and methodologies described in sections *4.1, 4.2 and 4.3 above* will be used while working in the *E-LEDA system*.

The Larman's method will be used as the base methodology for the development of the project since this is a lightweight methodology (even considered by some people as agile), that provides enough documentation for the continuity of project enhancing the maintainability and extensibility of the *E-LEDA system*.

The pomodoro technique will be used to discretize the working time lapses without interruptions (or at least, as few interruptions as possible) in order to provide a simpler understanding of the work that have been carried out (is easier to understand 15 time units than 7 hours 30 minutes, for large working intervals specially). The pomodoro technique provides also a frame for accurate task time estimation, by dividing the tasks in simpler tasks (*divide and conquer principle)*.

Merged with the pomodoro's discrete time lapses and the incremental construction phases of the Larman's method, PSP will be used as the base for improving the efficiency of the project in the different construction phases. A simplified version of PSP will be used to control the required amount of work for each task and would serve as a base for a deeper analysis on possible management weaknesses.

The merge of all those techniques intends to provide more accurate estimations on task effort and improve the efficiency. Using simplified methodologies should increase the productivity of the project and, by reducing the complexity of the described techniques, reduce the amount of time required to carry out the effort and efficiency analysis and soften the required learning curve.

5 Other Tools and Technologies

Many different tools have been used to carry out successfully the *E-LEDA project*, this section describes briefly the used tools and technologies that have not been explained in previous sections.

- **OmniPlan**(OmiGroup, 2012)**:** Software tool developed by the *OmniGroup,* for task planning, resource assignation and *Gannt diagram* representation, the main tool used to plan the project.
- **OmniGraffle** (OmniGroup, 2012)**:** Software tool developed by the *OmniGroup*, for diagram representation and prototyping. The project prototypes and diagrams are made with this tool using a collection of third party stencils.
- **Visual Paradigm** (Visual Paradigm, [s.d.])**:** Software tool for *UML* design, the *UML* diagrams of the project have been made with *visual paradigm*.
- **Microsoft Office** (Microsoft, 2013)**:** Suite developed by *Microsoft*, including various desktop applications for office works. The documentation of the project have been done using *MS Word,* for other purposes some of the other applications have also been used.
- **Git** (Git, [s.d.])**:** Free OpenSource control version system, used in the project for code versioning and source control.
- **MySQL** (MySQL, 2013)**:** Open Source rational database management system, used for the database management in the project.

6 Learning Management Systems (LMS)

The *E-LEDA system* is intended to work together with a Learning Management System (LMS). In words of Paulsen:

> *A Learning Management System is a broad term that is used for a wide range of systems that organize and provide access to online learning services for students, teachers, and administrators. These services usually include access control, provision of learning content communication tools, and organizations of user groups. Another term that often is used as a synonym to LMS is learning platform.*

(Paulsen, 2002)

The data for the *E-LEDA system's* analysis will be obtained from LMSs, some of the most popular ones are Moodle, Chamilo, Blackboard, ATutor or Sakai Project, the Table 4.4-1 LMS Features Comparison provides a small comparison between those learning management systems.

LMS	Software Requirements	SCORM	Free / Licence	Extensible	Authentication
Chamilo (Chamilo, 2013)	Windows, Linux, Mac	YES	YES / GNU/GPLv2	YES	LDAP, CAS, Shibboleth
Moodle (Moodle, 2013)	Windows, Linux	YES	YES / GNU-Trademark	YES	LDAP, Imap, NNTP
Blackboard (Blackboard, 2013)	Windows, Linux, Mac	YES	NO / Private-Trademark	YES	LDAP, CAS, Shibboleth
ATutor (ATutor, 2013)	Windows, Linux	YES	YES / OpenSource	YES	User-Password
Sakai Project (Sakai, 2013)	Windows, Linux, Mac	YES	YES / OpenSource	YES	LDAP

Table 4.4-1 LMS Features Comparison[7]

The *E-LEDA system* will focus in two of the most popular OpenSource LMS (Moodle & Chamilo), although the intention of the project is to provide in the future a universal system able to obtain data and communicate with most of the mayor LMS working nowadays.

[7] Data Source: (Mateos Carrera, 2011) & (Blackboard, 2013)

Planning and Requirements Specification

7 General Description and Objectives

The *E-LEDA system* objective is to provide lecturers using e-learning systems a tool for the analysis of the course suitability in the LMS. Using an iOS application the *E-LEDA system* is able to access the content of the lecturer's course in a LMS and display a series of charts indicating the progress of the students and the suitability of the tasks in a course.

The *E-LEDA system* is composed by three different sub-systems, and iOS application for the lecturer, a web application for the LMS administration and series of cloud services integrating the functionality between the other two *E-LEDA* sub-systems between each other and with the LMS. Figure 4.4-1 E-LEDA Architecture below represents the architecture of the *E-LEDA system* including the users, LMSs and the different sub-systems of the *E-LEDA system*.

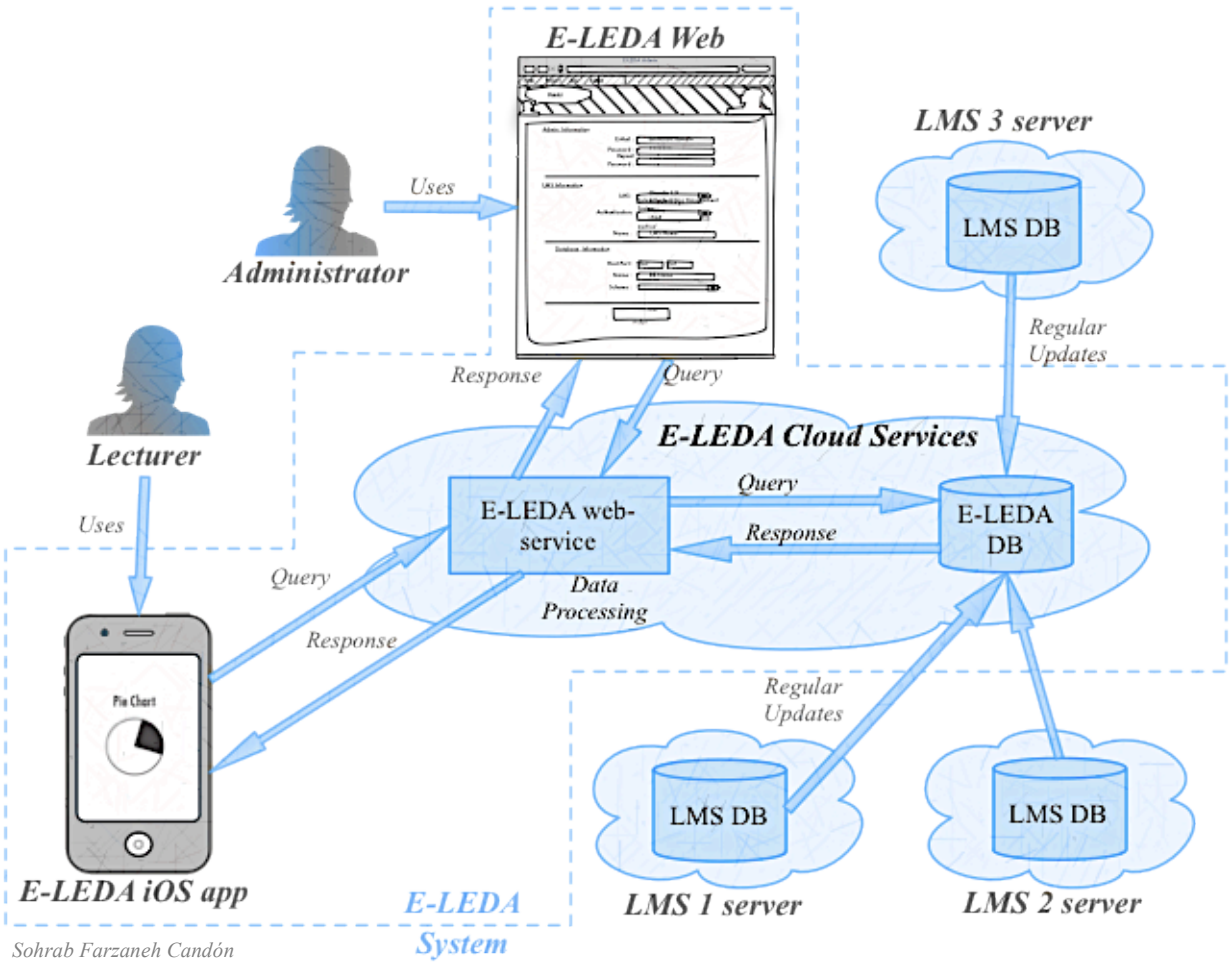

Figure 4.4-1 E-LEDA Architecture

A deeper description of the *E-LEDA system* and the *E-LEDA architecture* can be found in the *Specific Requirements Document* attached to this project (Farzaneh Candón, SRS, 2013).

8 Initial Draft Planning

This section describes the initial planning for the *E-LEDA system*, is important to notice the estimation component and the high level of uncertainty at this point of the project. It is also remarkable the lack of previous projects development using the methodologies and methods defined for the *E-LEDA project* which might considerably affect the precision of the initial estimation.

Table 4.4-1 Initial Plan Leading Tasks below represents the main tasks initially planned for the *E-LEDA project* including the initial and final date and the estimated effort. The time and effort are calculated using a 4 hours-a-day working calendar for each participant in the project. The calendar also takes into account the national holidays and days off.

Task	Starts	Ends	Effort (h)
Planning and Requirement Specification	26/09/2012	06/11/2012	101
Construction Phase 1	06/11/2012	06/12/2012	138
Construction Phase 2	06/12/2012	15/01/2013	138
Construction Phase 3	15/01/2013	20/02/2013	138
Installation	20/02/2013	26/02/2013	18
Final Delivery	-	26/02/2013	-

Table 4.4-1 Initial Plan Leading Tasks

It is important to remark that the number of construction phases is unknown in this phase of the project; therefor the number of construction phases is also estimated and may change along the *E-LEDA project* development. The Figure 4.4-1 Initial Plan General Gant Diagram below represents the Gantt diagram of the general initial plan including the tasks planned for the *planning and requirement specification* and *construction phase 1* task groups.

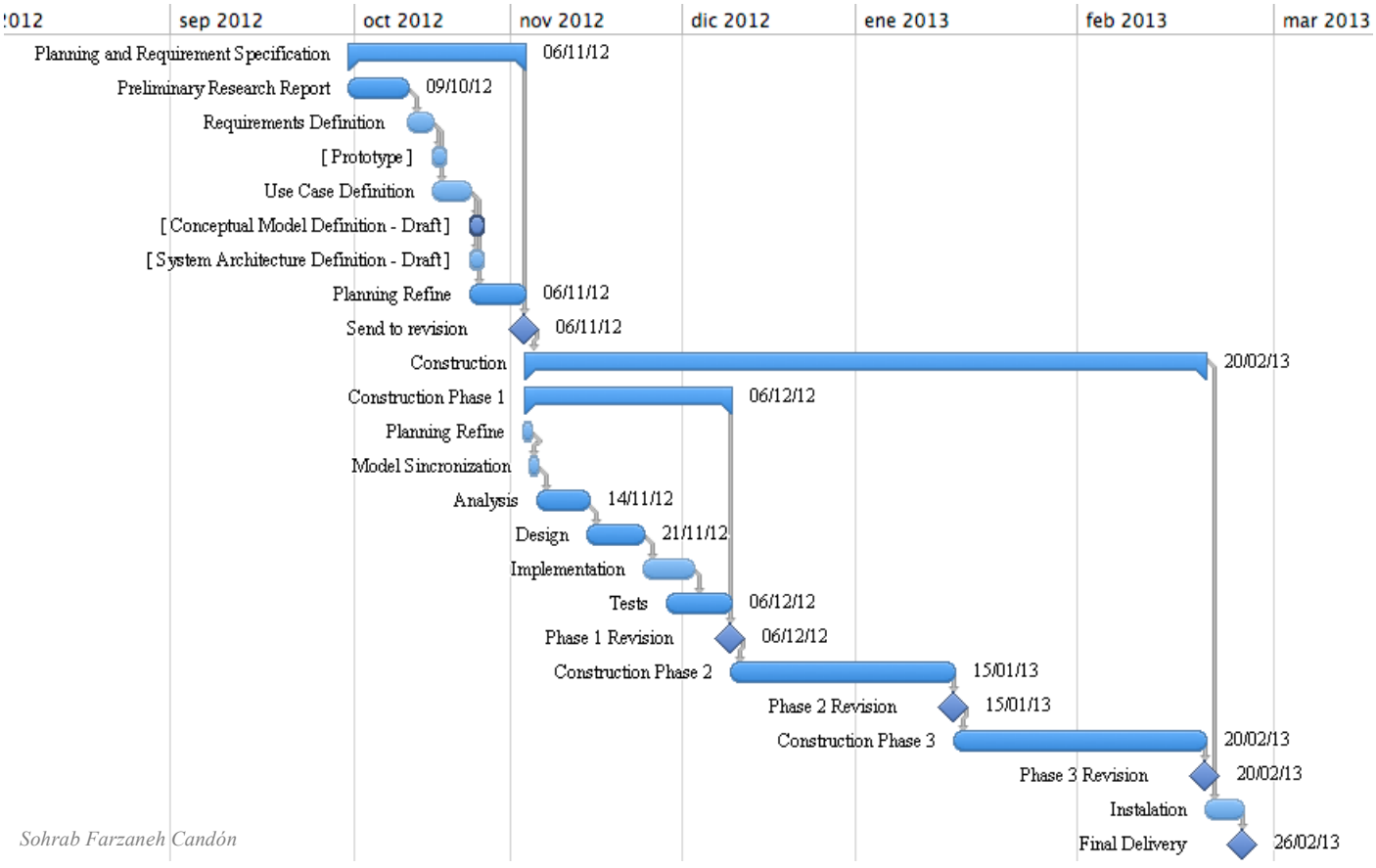

Figure 4.4-1 Initial Plan General Gant Diagram

The Figure 4.4-2 Initial Plan Construction Phase Gantt Diagram below includes the extended Gantt diagram of the construction phase 1, and serves as an example of the other construction phases planning.

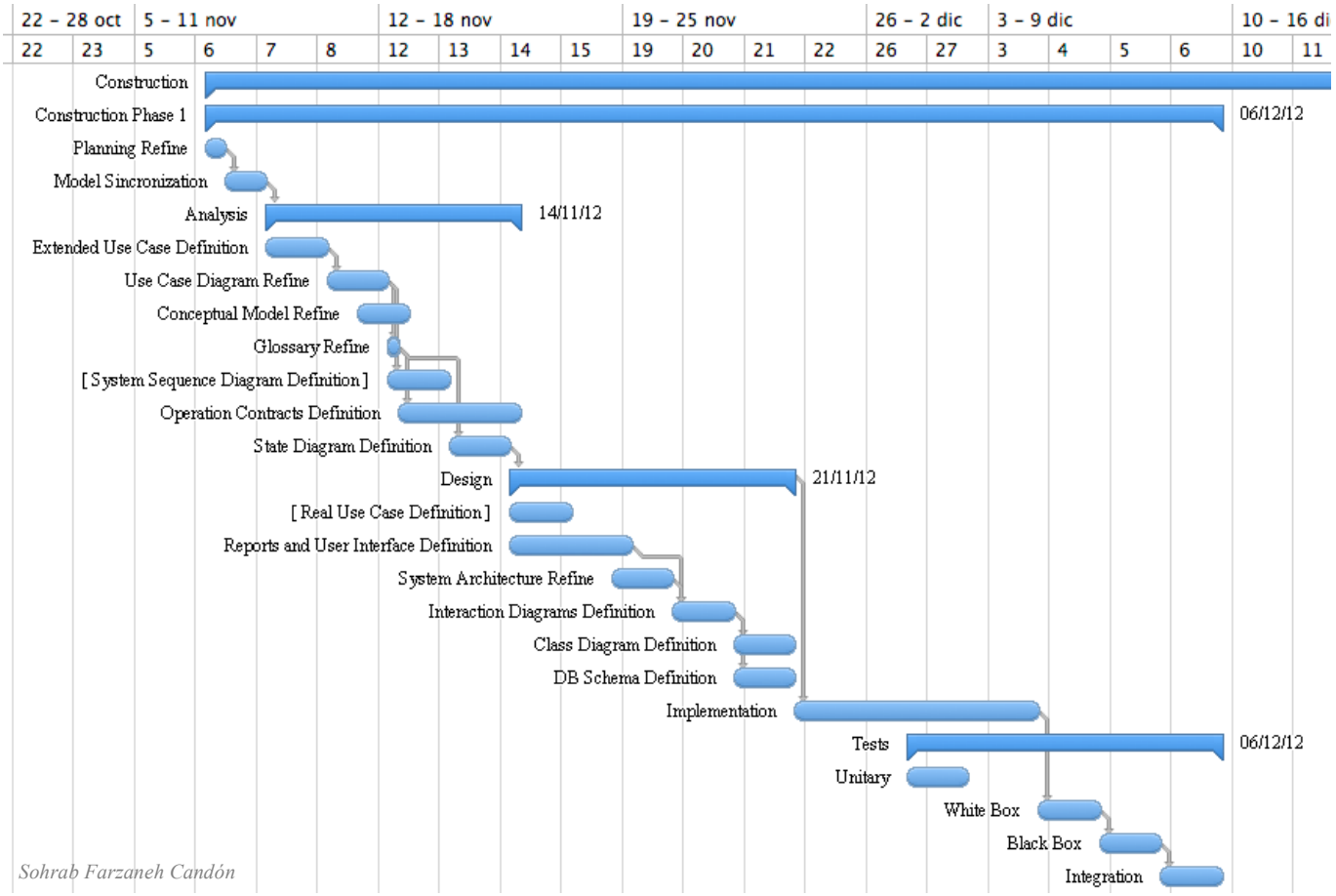

Figure 4.4-2 Initial Plan Construction Phase Gantt Diagram

9 Requirements Definition

The requirements definitions of the *E-LEDA system* are collected in the *Software Requirements Specification* included as an extra document in the *E-LEDA project* (Farzaneh Candón, SRS, 2013).

10 Use Case Definition

This section defines a general overview of the actions defined for each of the *E-LEDA system's* sub-systems. As defined in *section 7 above* the *E-LEDA system* is divided in three sub-systems: iOS, Web and Cloud Services (CS).

The Figure 4.4-1 iOS Sub-System Use Cases defines the specific use cases for the *iOS sub-system*. The *iOS sub-system* includes five different action groups (represented as packages in the figure): Authentication, Course Information, Student Information, Task Information and Chart Properties.

The Figure 4.4-2 Web Sub-System Use Cases include the specific use cases for the *Web sub-system*. It defines two use case groups: Authentication and LMS management.

The Figure 4.4-3 CS Sub-System Use Cases specifies the use cases for the *CS sub-system* including three groups of functionalities: Authentication, LMS Management and Data Analysis.

The detailed use case definitions of the *E-LEDA system* are collected in the *Use Case Specification* included as an extra document in the *E-LEDA project* (Farzaneh Candón, UCS, 2013).

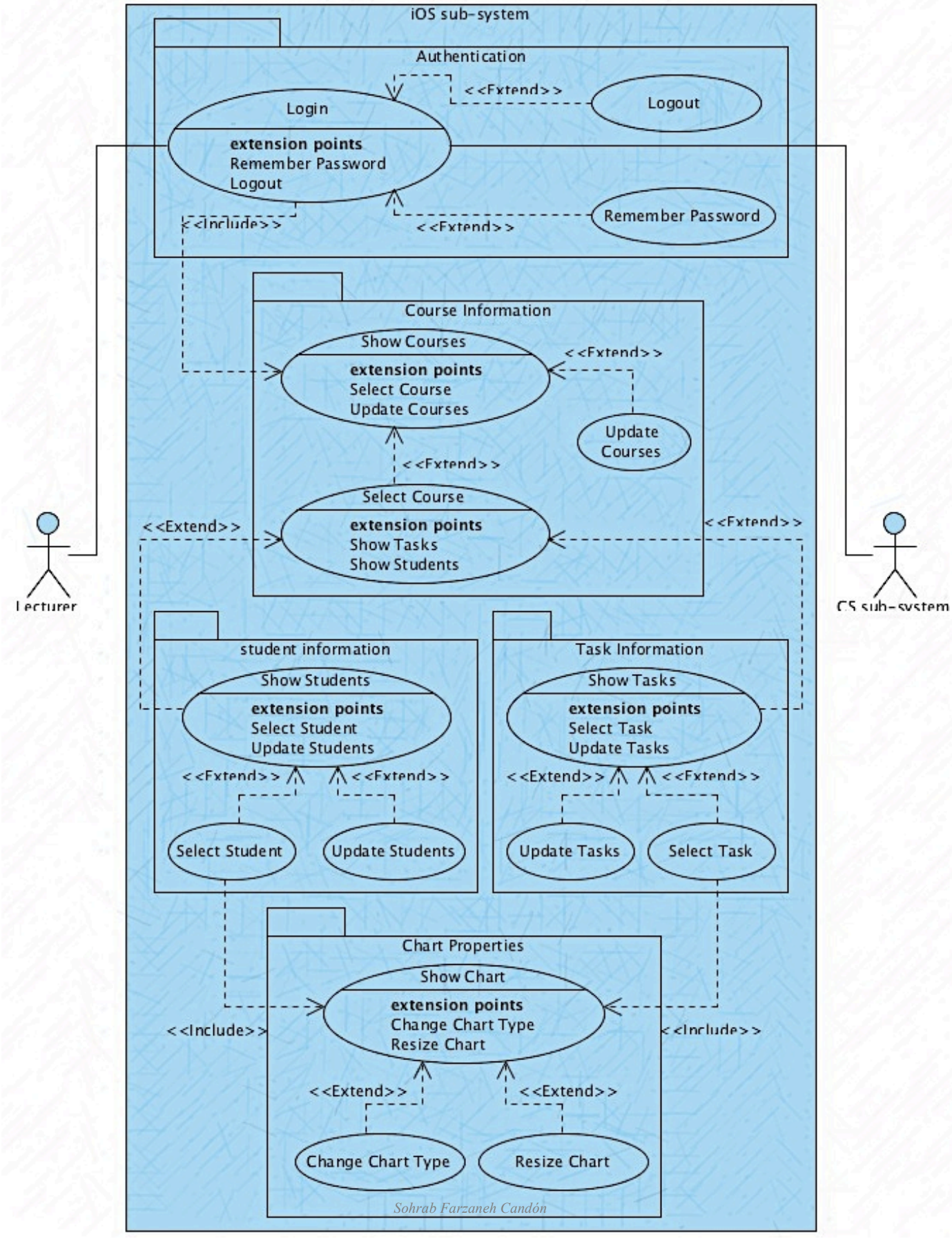

Figure 4.4-1 iOS Sub-System Use Cases

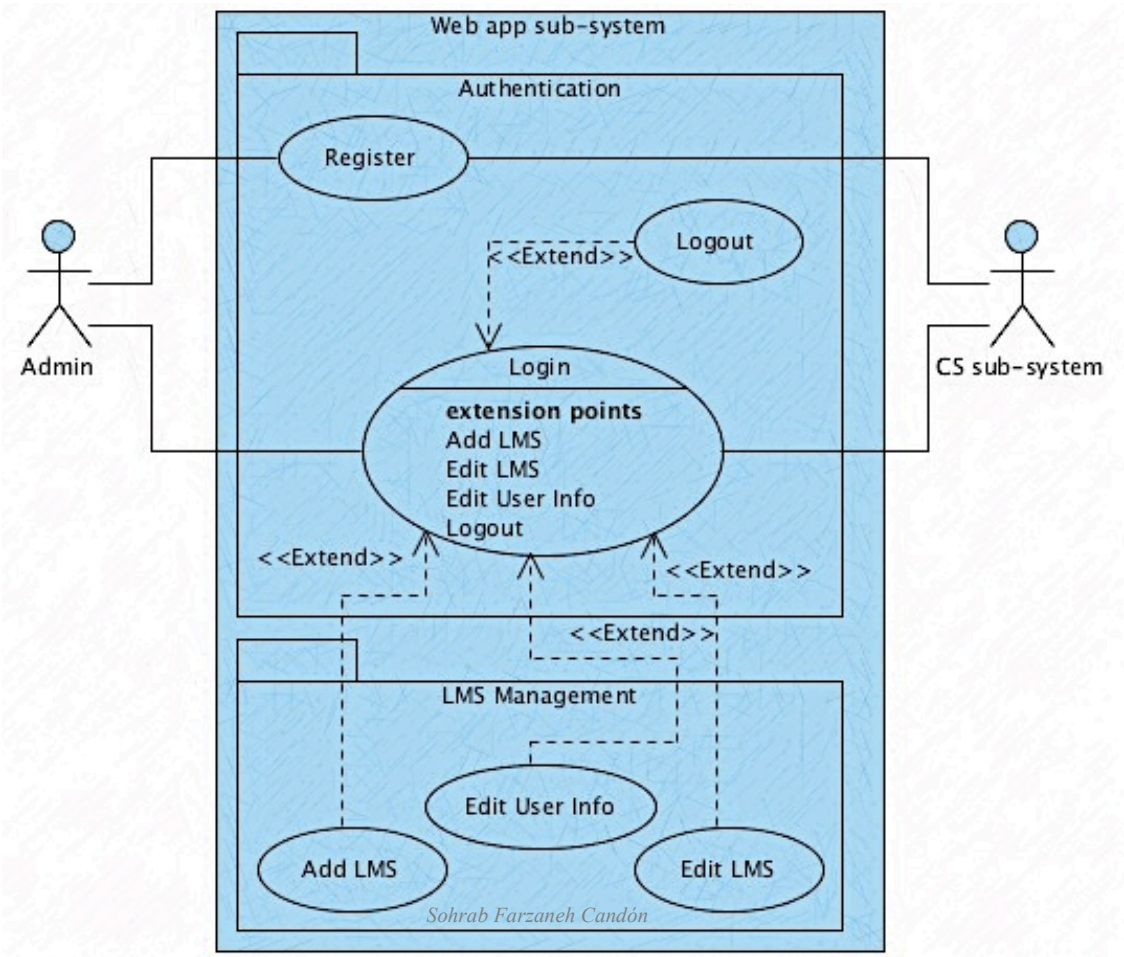

Figure 4.4-2 Web Sub-System Use Cases

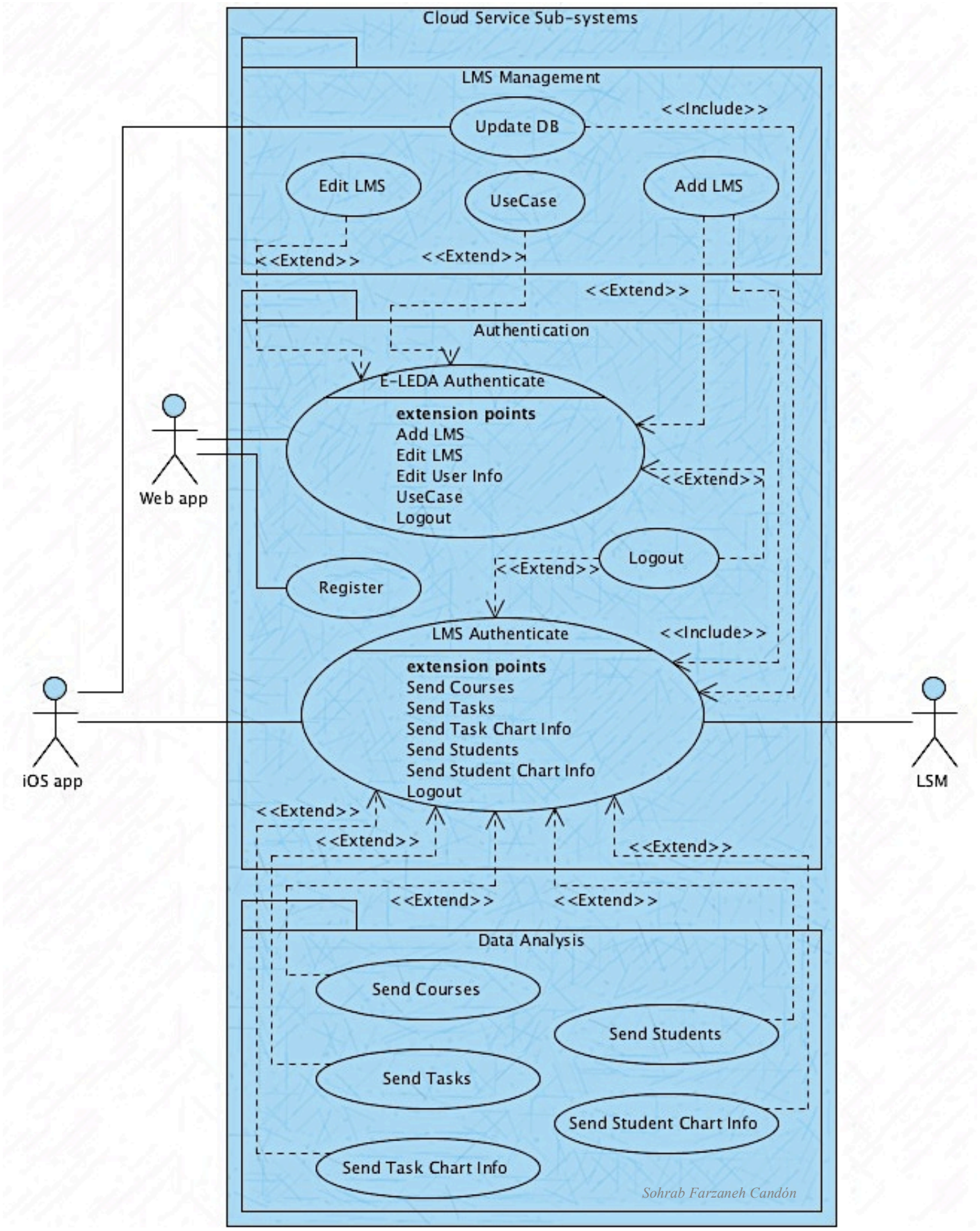

Figure 4.4-3 CS Sub-System Use Cases

11 Construction Phases

Following the Larman's method once the initial use case have been designed the project is divided in different incremental iterative construction phases. In order to check the correctness and completeness of the use cases the traceability between the use cases and the requirements shall be checked.

11.1 Traceability Matrixes

This section represents the traceability matrix between the requirements and the use cases of the *E-LEDA system*. The following sections describe the traceability for the use cases with the functional and non-functional requirements for each sub-system of the *E-LEDA system*. The name used in the following tables corresponds with the identifiers given to the requirements and use cases, the complete description of the requirements can be found in the *Software Requirement Specification* (Farzaneh Candón, SRS, 2013). For a complete description of the use cases the *Use Case Specification* (Farzaneh Candón, UCS, 2013) document can be consulted.

11.1.1 Functional Requirements

All functional requirements shall correspond with at least one use case, and one use case should refer to at least one functional requirement. The following tables Table 11.1-1, Table 11.1-2 and Table 11.1-3 represent this relationship in the different sub-systems of the *E-LEDA system*.

UC / FR	iOS001	iOS002	iOS003	iOS004	iOS005	iOS006	iOS007	iOS008	iOS009	iOS010	iOS011	iOS012	iOS013	iOS014	iOS015
IFR001	X														
IFR002			X												
IFR003				X											
IFR004					X										
IFR005								X							
IFR006									X						
IFR007										X					
IFR008											X				
IFR009											X				
IFR010											X				
IFR011						X									
IFR012							X								
IFR013										X					
IFR014											X				
IFR015											X				
IFR016											X				
IFR017											X				
IFR018												X			
IFR019													X	X	X
IFR020		X													

Table 11.1-1 iOS sub-system traceability matrix

FR \ UC	WEB001	WEB002	WEB003	WEB004	WEB005	WEB006
WRF001	X					
WRF002		X				
WRF003					X	
WRF004				X		
WRF005			X			
WRF006						X

Table 11.1-2 Web sub-system traceability matrix

FR \ UC	CS001	CS002	CS003	CS004	CS005	CS006	CS007	CS008	CS009	CS010	CS011	CS012	CS013
CFR001			X										
CFR002			X										
CFR003		X											
CFR004					X								
CFR005				X									
CFR006						X							
CFR007							X						
CFR008								X					
CFR009									X	X			
CFR010											X		
CFR011											X		
CFR012										X			
CFR013	X												
CFR014												X	
CFR015													X

Table 11.1-3 Cloud Service sub-system traceability matrix

11.1.2 Non-Functional Traceability Matrix

The non-functional requirements might correspond to a use case and a use case might correspond to a non-functional use case. The Table 11.1-4 Non-functional traceability matrix represents this relationship

FR \ UC	UIR005	SIR001	SIR002	CIR002	DCR002	DCR003	NFSR001	NFSR002	NFR003	NFR004
iOS003								X	X	X
iOS013	X									
iOS014	X									
iOS015	X									
iOS020										
WEB001								X		
CS001								X		
CS003			X	X	X	X				
CS004					X	X				
CS005		X			X	X				
CS006		X								
CS013							X			

Table 11.1-4 Non-functional traceability matrix

11.2 Prioritisation

In order to define the different construction phases of the *E-LEDA project*, and following the Larman's method, a prioritisation of use cases is to be done. The use cases prioritising is done by taking into account the priority and necessity assigned to each functional requirement and the traceability with each use case.

Using this prioritisation metrics the functional requirements could be divided in four groups:

- **Necessity:** Essential **Priority:** High
- **Necessity:** Essential **Priority:** Medium
- **Necessity:** Desirable **Priority:** High
- **Necessity:** Desirable **Priority:** Low[8]

The Table 11.2-1 Strict Use Case Prioritisation below include the strict prioritisation taking into account only the given functional requirement priority and necessity.

Essential High	**Essential Medium**	**Desirable High**	**Desirable Medium**
iOS004-Show Course	iOS001-Login (Lecturer)	WEB003-Logout(Admin.)	iOS003-Remember Password
iOS005-Select Course	iOS002-Logout (Lecturer)	WEB006-Edit User Info.	iOS011-Change Chart type
iOS006-Show Students	WEB001-Register Admin.	CS006-Edit LMS	iOS012-Resize Chart
iOS007-Select Student	WEB002-Login (Admin.)	CS012-Edit User Info.	iOS013-Update Courses
iOS008-Show Tasks	WEB004-Logout (Admin.)		iOS014-Update Students
iOS009-Select Task	WEB005-Add LMS		iOS015-Update Tasks
iOS010-Show Chart	CS001-Register		
iOS010-Show Chart	CS002-Auth. (E-LEDA)		
CS007-Send Courses	CS003-Auth. (LMS)		
CS008-Send Students	CS004-Update DB		
CS009-Send Tasks	CS005-Add LMS		
CS010-Send Student Chart Info	CS013-Logout		
CS011-Send Task Chart Info			

Table 11.2-1 Strict Use Case Prioritisation

In order to balance the workload and develop the construction phases taking into account also the semantic relationship between the different use cases, based on the division of the requirements, the amount of requirements of each group and the semantic relation between them, four construction phases will be finally planned.

- The *first construction phase (Basic app)* includes the required use cases for the basic iOS application (including the required cloud services).
- The *second construction phase (Authentication)* is focused in the user authentication including both LSM authentication and E-LEDA authentication.
- The *third construction phase (Administrator functionality)* intends to develop the administrator functionalities including the web applications.
- The *fourth construction phase (Complete App)* is aimed to develop the complete iOS application including all necessary charts.

The Table 11.2-2 Final Use Case Prioritisation below represents the use cases in each of the four final construction phases.

[8] Desirable-Medium also exists, but is irrelevant in this case, because the only use case affected is iOS011, and most of its related functional requirements are Desirable-Low

CP1 Basic App	CP2 Authentication	CP3 Admin. Functionality	CP4 Complete App
iOS001-Login (Lecturer)[9]	iOS001-Login (Lecturer)	WEB003-Logout(Admin.)	iOS003-Remember Password
iOS004-Show Course	iOS002-Logout (Lecturer)	WEB004-Add LMS	iOS011-Change Chart type
iOS005-Select Course	WEB001-Register Admin.	WEB005-Edit LMS	iOS012-Resize Chart
iOS006-Show Students	WEB002-Login (Admin.)	WEB006-Edit User Info	iOS013-Update Courses
iOS007-Select Student	CS001-Register	CS005-Add LMS	iOS014-Update Students
iOS008-Show Tasks	CS002-Auth. (E-LEDA)	CS006-Edit LMS	iOS015-Update Tasks
iOS009-Select Task	CS003-Auth. (LMS)	CS012-Edit User Info.	
iOS010-Show Chart	CS004-Update DB		
CS003-Authenticate (LMS)[9]	CS005-Add LMS		
CS007-Send Courses	CS013-Logout		
CS008-Send Students			
CS009-Send Tasks			
CS010-Send Student Chart Info			
CS011-Send Task Chart Info			

Table 11.2-2 Final Use Case Prioritisation

12 Planning Refine & Budget

12.1 Planning Refine

Once the system is well defined including detailed requirements and use cases and divided in incremental iterations, the planning of the *E-LEDA system* can be refined providing a more accurate estimation of the size and effort required for the *E-LEDA project* development.

Table 12.1-1 Refined Plan Leading Tasks specifies the planned starting date, ending date and effort for each of the mayor task group of the *E-LEDA system* taking into account the estimated complexity of each of the construction phases defined in *section 11.2 above*

Task	Starts	Ends	Effort (h)
Planning and Requirement Specification	26/09/2012	21/02/2013	129,5
Construction Phase 1 – iOS Basic app	27/02/2012	15/04/2013	119
Construction Phase 2 - Authentication	15/04/2013	27/06/2013	320
Construction Phase 3 – Admin. Functionality	27/06/2013	15/08/2013	210
Construction Phase 3 – iOS Complete app	15/08/2013	10/10/2013	240
Installation	10/10/2013	14/10/2013	18
Final Delivery	-	14/10/2013	-

Table 12.1-1 Refined Plan Leading Tasks

The Figure 12.1-1 Refined Plan General Gantt Diagram shows the Gantt diagram for the main tasks of the refined planning in the *E-LEDA project.* In the Gantt diagram is easy to appreciate the gap in the middle of to of the main tasks *(requirements definition and use case definition)* these gaps were caused by an unexpected reorganization of the human resources due to an urgent project delivery.

Due to time restrictions and the amount of effort required for the *E-LEDA system*, by the time of presenting this thesis only the *construction phase 1* will be carried out. The rest of *E-LEDA project* construction phases will be developed in the future. Another software engineer might develop the rest of the construction phases of the E-LEDA system.

The *Construction Phase 1* document (Farzaneh Candón, CP1, 2013) includes the detail of the refined plan for the construction phase 1.

[9] Only partial implementation, just the needed functionality to allow the rest of the use cases to work

The full updated planning information of the *E-LEDA project* can be consulted in the on-line resource http://seldata.sel.inf.uc3m.es/people/sohrab/eleda/planning[10]. This resource includes the comparison between the refined plan and the real time spent in the project.[11]

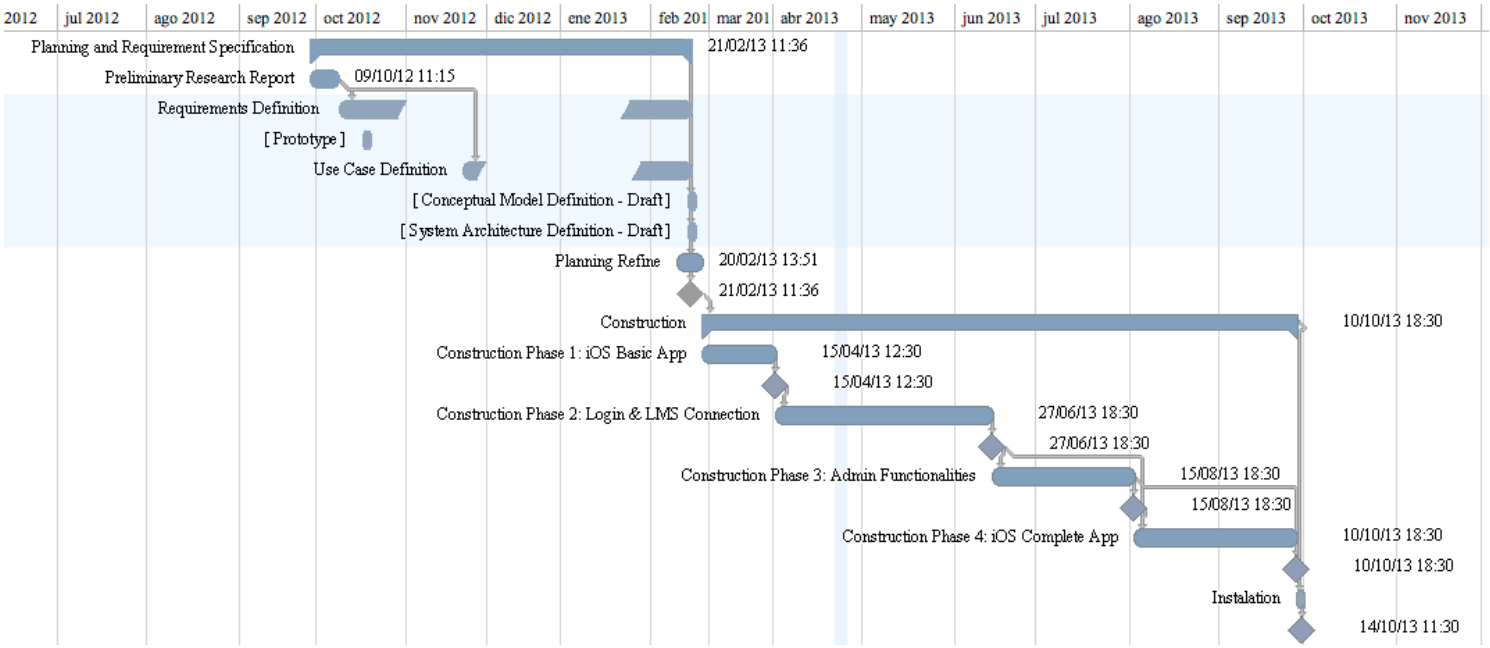

Figure 12.1-1 Refined Plan General Gantt Diagram

12.2 Budget

This section describes the budget for the *E-LEDA system,* including the description of the indirect costs and expected benefit, this budget is not intended to be shown to any client since the benefit and indirect cost percentages may vary depending on the situation and the environment of the project in order to adjust the price as much as possible to the founding requirements. All costs are shown excluding VAT except the final cost of the project.

The *E-LEDA system Budget* is divided in four tables including different calculations related to the costs of human resources, equipment and other related costs, and one summary table including the net costs of the different budget areas.

Table 12.2-1 Human Resources Cost represents the cost of the different human resources involved in the *E-LEDA project*. As explained in *section 12.1 above* Sohrab Farzaneh Candón will be the main junior software engineer until the end of the *construction phase 1* is accomplished, after that moment another junior software engineer might continue the project, although the name of this new engineer is unknown by the time of writing this document.

Table 12.2-2 Equipment Cost describes the cost of the needed equipment (hardware and software) including a depreciation period of 60 months.

Table 12.2-3 Other Direct Costs include the cost of other direct costs related with the *E-LEDA system*.

Table 12.2-4 Cost Summary includes the net summary of the sub-total cost of each of the different calculations needed for the budget elaboration of the *E-LEDA system*.

Table 12.2-5 Final Budget represents the final budget of the *E-LEDA system* including the total cost of the *E-LEDA project*, the percentage calculated for indirect costs, the benefit and VAT.

[10] Some contents of the web site might be in *Spanish* due to OmniPlan's regional configuration. The cost appearing in the web site do not reflect the real cost of the project, it only takes into account human resources salaries.

[11] This resource is not a real-time resource. By the time of writing this document the last update was made on 25/04/2013

Last name, Name	Category	Dedication (months)	Cost (per month) (€)	Cost (total) (€)
Mora Soto, José Arturo	Senior Engineer	0,25	4.289,54	1.072,39
Farzaneh Candón, Sohrab	Engineer	6	2.694,39	16.166,34
Unknown	Engineer	6	2.694,39	16.166,34
	Total Duration	**12,25**	**Sub-Total Cost**	**33.405,07**

Table 12.2-1 Human Resources Cost

Description	Cost (€)	Dedicated to project (%)	Dedication (months)	Depreciation Period (months)	Imputable Cost (€)
iMac 21" 2,9 GHz (standard Configuration)	1.305,00	100	12,25	60	266,44
Dell E-series E1713S (17") monitor	119,00	100	12,25	60	24,30
OmniPlan 2.2.4	136,47	100	12,25	60	27,86
OmniGraffle Professional 5.4.2	142,39	100	12,25	60	29,07
Visual Paradigm for UML Modeller 9.0	84,26	100	12,25	60	17,20
Office 2008 for Mac	390,57	100	12,25	60	79,74
				Sub-Total Cost	**444,61**

Table 12.2-2 Equipment Cost

Description	Provider	Imputable Cost (€)
1 Year Hosting 1Gb (MySQL, PHP…)	nosolored	40,00
	Sub-Total Cost	**40,00**

Table 12.2-3 Other Direct Costs

Budget Areas	Total Costs (€)
Human Resources	33.405,00
Equipment Amortization	444,61
Subcontracting	0,00
Functioning Work	40,00
Sub-Total Cost	**33.889,68**

Table 12.2-4 Cost Summary

Charge type	Percentage Value	Monetary Value (€)
Indirect Costs	20%	6.777,94
Benefit	15%	6.100,14
VAT	21%	9.821,23
	Total Cost	**52.867,90**

Table 12.2-5 Final Budget

Construction Summary

13 Construction Phase 1: iOS Basic App

The objective of the first construction phase in the *E-LEDA project* is to deliver a working iOS app with the basic functionality, including *E-LEDA database* access, communication with the *CS sub-system*, and presentation of courses, students, tasks and pie charts in the *iOS app.*

For simplicity and flexibility the different construction phases will be delivered in separate documents following the rules and formats of this document. As explained in *section 12 above* the delivery of this thesis will only include the first construction phase of the *E-LEDA project.*

The detailed information about the *construction phase 1* can be consulted in the document *Construction Phase 1: iOS Basic App* attached to this project (Farzaneh Candón, CP1, 2013). The *construction phase 1* document includes the information about the design, implementation and tests of the *iOS Basic App.*

Conclusions

The *E-LEDA system* was designed and developed[12] using a combination of techniques (*Larman's method, simplified PSP and Pomodoro)* in order to improve the efficiency and quality of the work process. In this conclusions the results of the registered times and data will be presented in order to check the suitability of the software engineering techniques used in the *E-LEDA Project.*[13]

14 Time Distribution

Figure 12.2-1 Activity Time Share represents the amount of time worked in each of the activities composing the *E-LEDA project* including analysis, design, implementation, test, planning, research, documentation and other activities such as meetings for example.

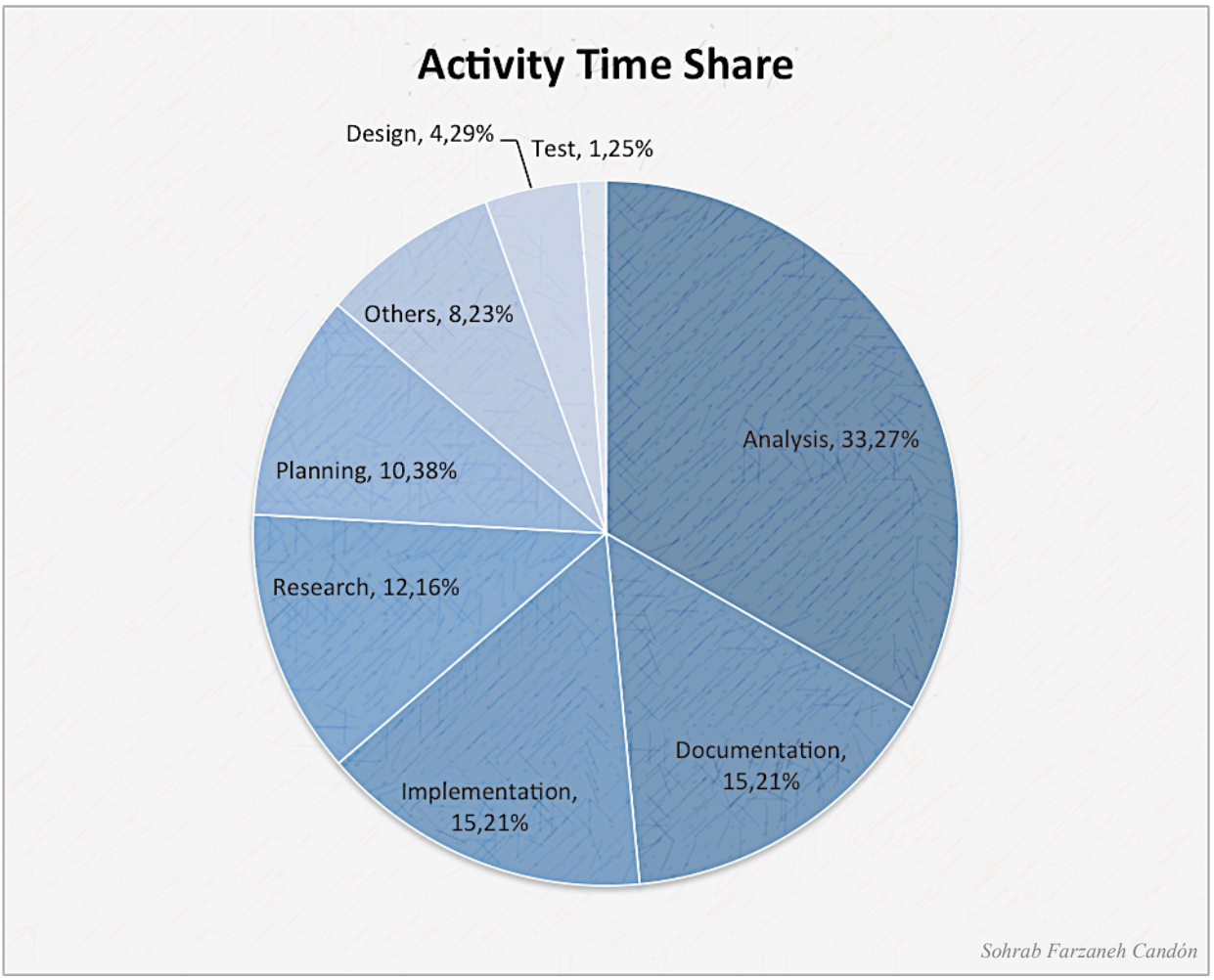

Figure 12.2-1 Activity Time Share

[12] Only construction phase 1 was developed, and the statistics only refer to that construction phase.
[13] The source data for all the shown charts can be consulted in PSP files:
http://seldata.sel.inf.uc3m.es/people/sohrab/eleda/psp/WeekActivityEstimation.htm
http://seldata.sel.inf.uc3m.es/people/sohrab/eleda/psp/WeekActivitySummary.htm
http://seldata.sel.inf.uc3m.es/people/sohrab/eleda/psp/ErrorsNotebook.htm
http://seldata.sel.inf.uc3m.es/people/sohrab/eleda/psp/TimeRecords.htm

The biggest activity in the *E-LEDA project* was clearly the *Analysis* of the *E-LEDA system*, this fact is easily explainable due to the fact that the hole *E-LEDA system* has been analysed and only the *first construction phase* has been designed, implemented and tested.

Documentation and implementation involve, each, around *15,2%* of the time dedicated to the project, again it is important to take into account that only the *first construction phase* has been developed. Another important consideration is the lack of knowledge of the programming languages (Objective-C, PHP) used in the *E-LEDA system* before the realization of the *E-LEDA project.*

Planning results are surprising due to the fact that since *January 2013* the planning technique changed using micro-task planning[14], which involves daily and weekly planning, the expected planning time was bigger than the final result obtained.

Design and testing times are also surprising. For the design, one more time it is important to remark that only the *first construction phase* was designed, even tough the designed system was only a portion of the *E-LEDA project* the value seems considerably low. This fact can be easily explained taking into account the philosophy of *Larman's method*. Larman's method is an incremental-iterative use case driven method, this means that the use cases of the system guides the design of the project, in this particular case and in order to provide the clearest possible description of the project and the maximum grade of extensibility, the analysis use cases where made as complete as possible (always trying not to interfere with the design decisions), thus the design were much guided and less time were needed for the *E-LEDA system's* design.

The detailed analysis and design explains also the unexpected result of the test time, due to the detailed description of what the system shall do, and the simplicity of the implementation, the quality of the final product was as good as intended failing only in two tests and correcting the errors in one hour.[15]

15 Activities Distribution Along the *E-LEDA project*

This section describes the distribution of activities over the time that the *E-LEDA project* lasted (taking into account that only the first *construction phase* has been developed).

Figure 12.2-1 Activities Estimation (Month Granularity) represents the estimation of the different activities of the project. As expected implementation covers the last period of the project while analysis is done the first and middle periods. An odd value might be the research time in the last quarter of the project development, this research is very comprehensible if we take into account that the developer has not programmed in objective-c or PHP before, so this research is exclusively about the programming languages needed for the development.

Another unexpected value might be the depression while approaching December, this is due to the fact that there was another project delivery in January, and the human resources had to be moved to the other project due to the imminent deadline.

Figure 12.2-2 Activities Real Time (Month Granularity) represents the real time spent in each of the activities along the time with a monthly granularity. The differences between the estimations and the real values can be observed comparing the charts. Figure 12.2-2 Activities Real Time (Month Granularity) shows a more continuous shape than the estimation, but with lower values.

Figure 12.2-3 Activities Estimation (Week Granularity) and Figure 12.2-4 Activities Real Time (Week Granularity) represents the same information than the charts with month granularity but obtaining the data weekly instead of monthly, in this case the most remarkable fact is the difference between the two estimation periods. The first estimation period *(September – December)* show a much more continuous shape than the second estimation period *(January – April)*, this is because in the second estimation period micro-planning was used, which forces the daily and weekly small tasks planning, leading into a better understanding of the daily situation an allowing to plan specifically for smaller time periods[14].

[14] For more information about micro-planning see *section 16* Estimation Accuracy And Micro-Planning
[15] The test results can be consulted in *Construction Phase 1: iOS Basic App* (Farzaneh Candón, CP1, 2013).

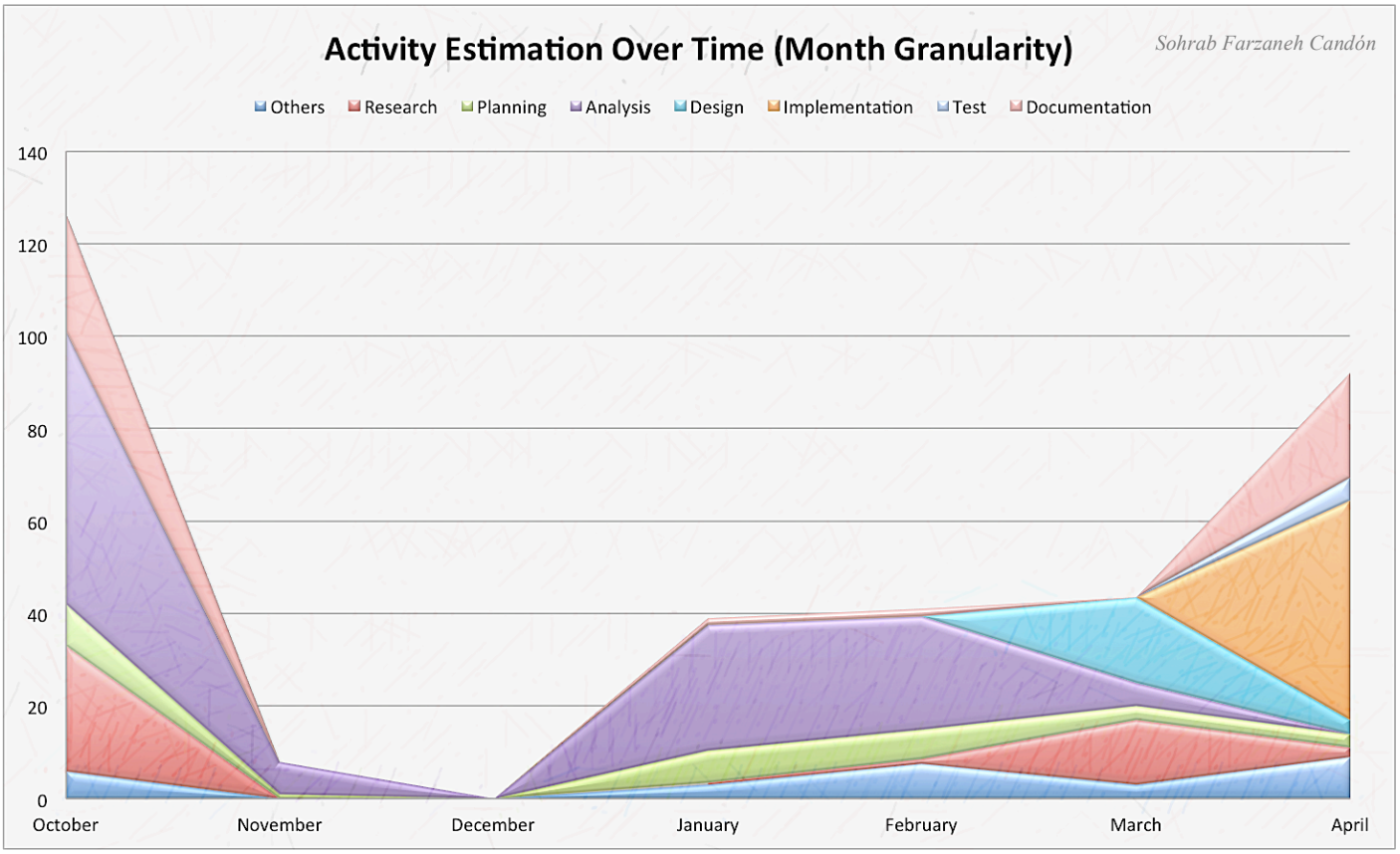

Figure 12.2-1 Activities Estimation (Month Granularity)

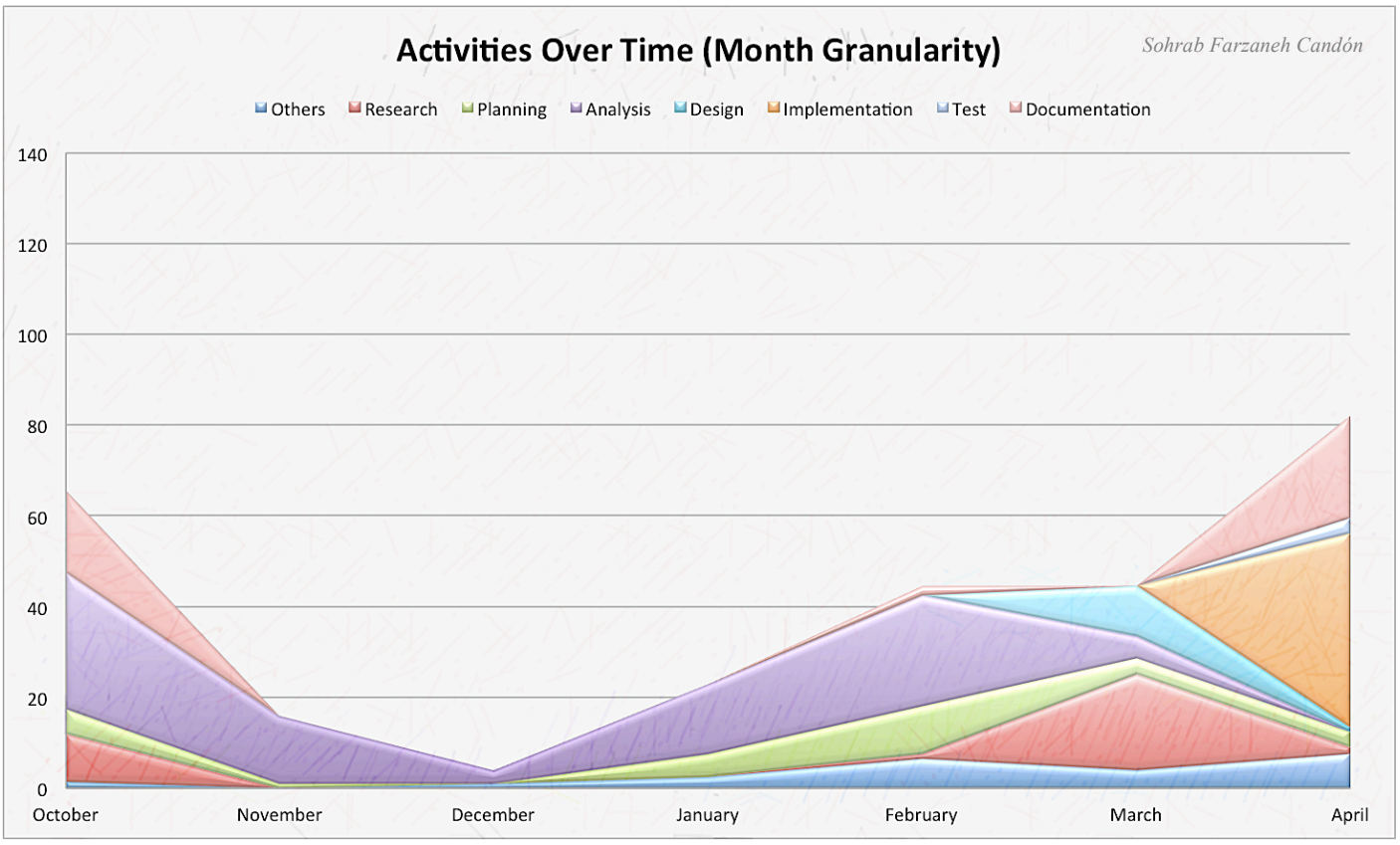

Figure 12.2-2 Activities Real Time (Month Granularity)

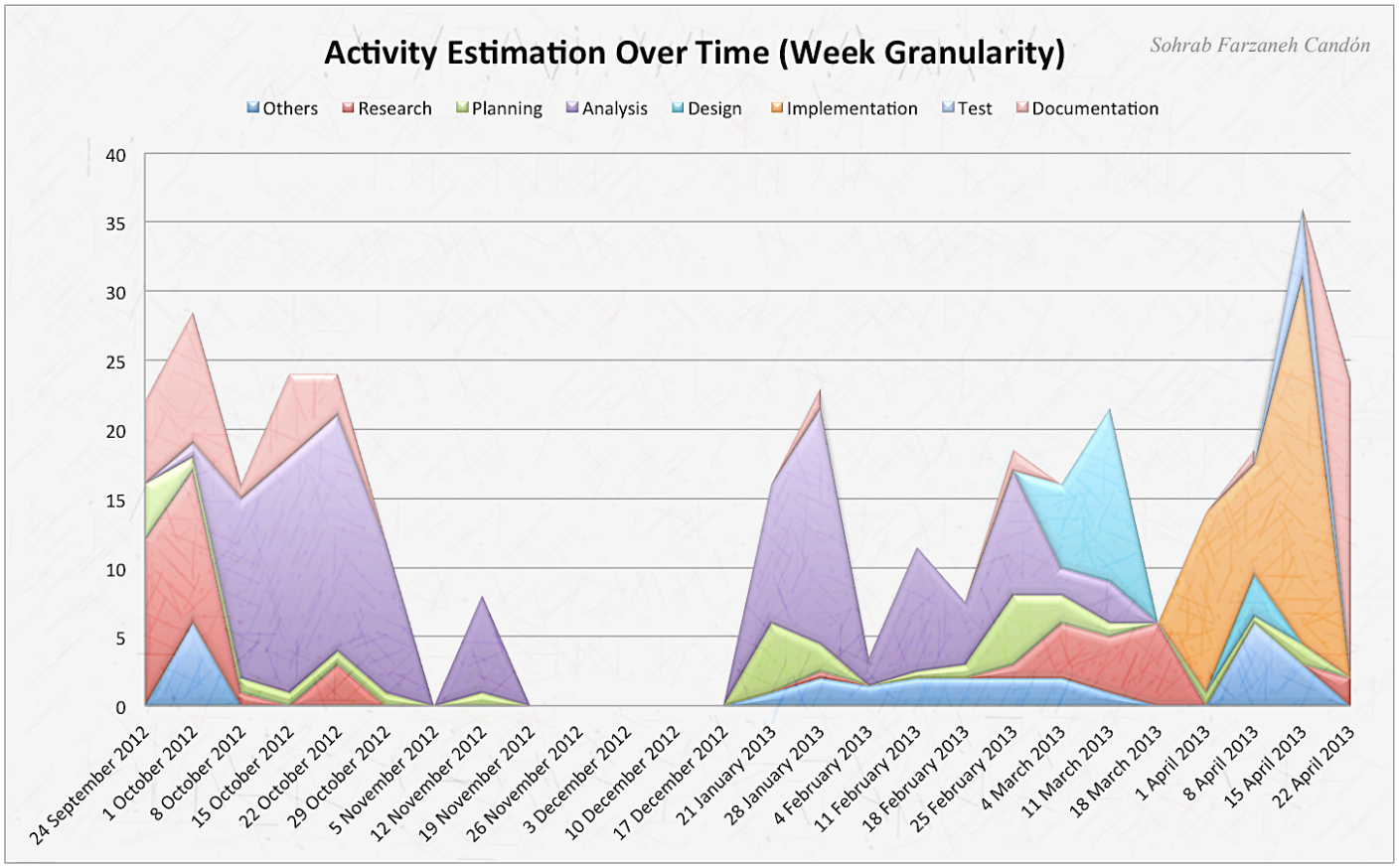

Figure 12.2-3 Activities Estimation (Week Granularity)

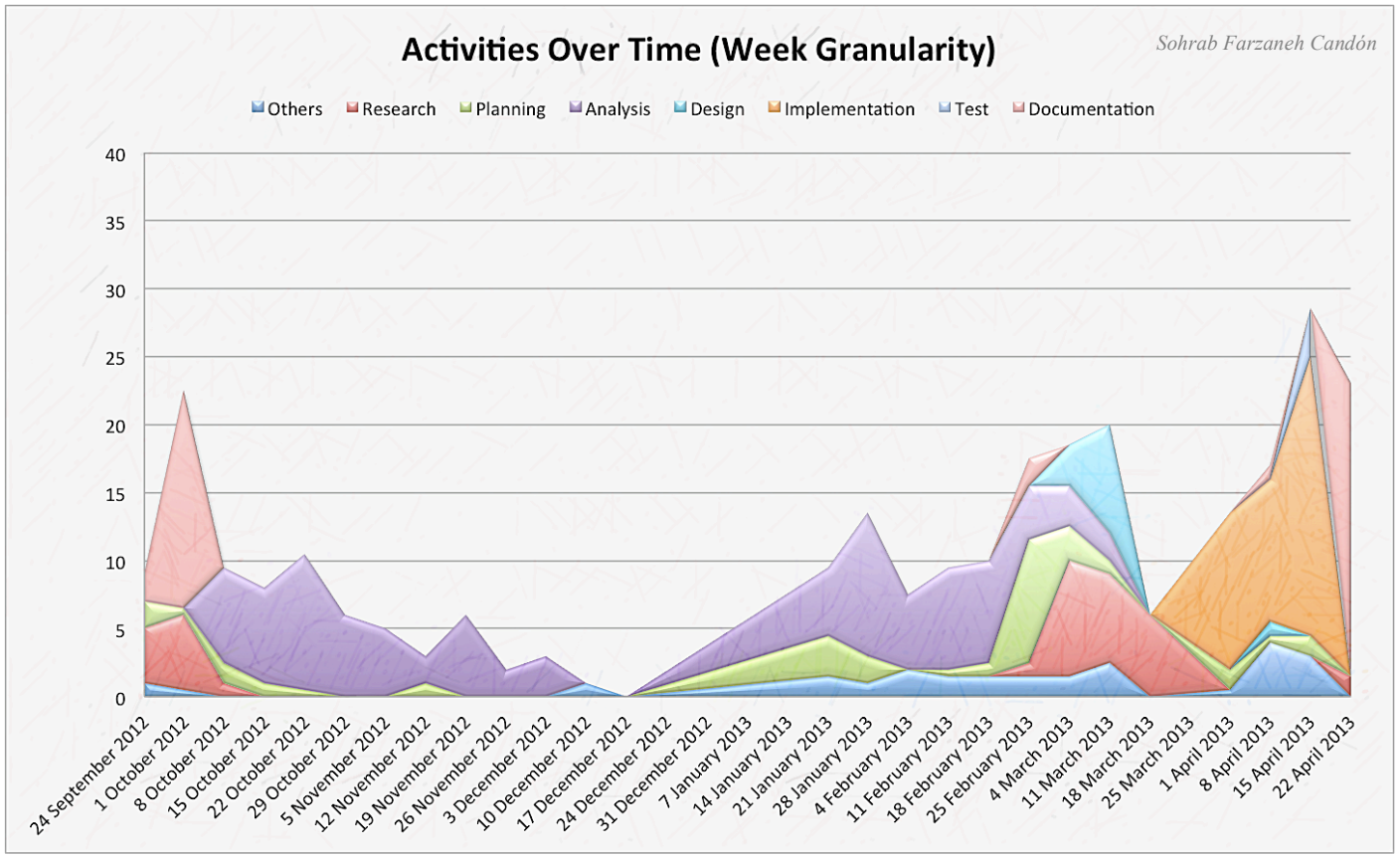

Figure 12.2-4 Activities Real Time (Week Granularity)

16 Estimation Accuracy And Micro-Planning

While developing the *E-LEDA system* many different tools have been used in order to improve the efficiency, quality and extensibility of the *E-LEDA project (see section 4 above)*. One of the resulting products of the *E-LEDA project* was a mixture of techniques that evolved along with the project development, the final working method in the *E-LEDA project* is a combination between Larman's method following the incremental-iterative use case driven lifecycle, a simplification of PSP which allowed to obtain and analyse the working times and estimations and the pomodoro technique which allowed to discretise the time units into pomodoros and establish the micro-task planning (or micro-planning).

The PSP technique is intended to measure the efficiency while programming, taking into account the planning, design, implementation and test phases. In the case of the *E-LEDA system*, the time was measured from planning to test, including also analysis, design, implementation, documentation, research, and other activities such as communication time. Due to the nature and simplicity of the implementation phase[12 above]there are PSP measure that have not been taken into account, such as the compilation time or the lines of code, therefor not all the PSP tools have been used, only week activity estimations, time records, seminal activity summary, and error notebook have been taken from PSP for the *E-LEDA system* development.

While the project was being developed, the working technique was evolving and the different techniques were merging more suitably, thus the project can be divided in two clearly different phases, the first half of the project was estimated using previous experiences and following the Larman's method given tasks, the last half included micro-task planning as an extension of the previous planning method[13].

Micro-task planning is the combination between conventional task planning and Pomodoro technique task planning. The conventional planning follows previous experiences, methodologies and other techniques to plan from small to large-scale projects in time using a series of task. Pomodoro planning forces the division of tasks lasting longer than 6-7 pomodoros (3 – 3,5 hours) in several task using the divide and conquer principle.

While using the pomodoro technique along with the conventional planned task, conventional planned tasks are usually divided in several small sub-tasks (shorter than 3,5 hours). The task division is not made at the very first plan, but daily or weekly taking into account the real working time available.

Figure 12.2-1 Estimation Accuracy (Conventional Planning) represents the estimation and real time used in the first part of the *E-LEDA project* development using conventional planning. Figure 12.2-2 Estimation Accuracy (Micro-Planning) represents the estimation and real time using the micro-planning technique.

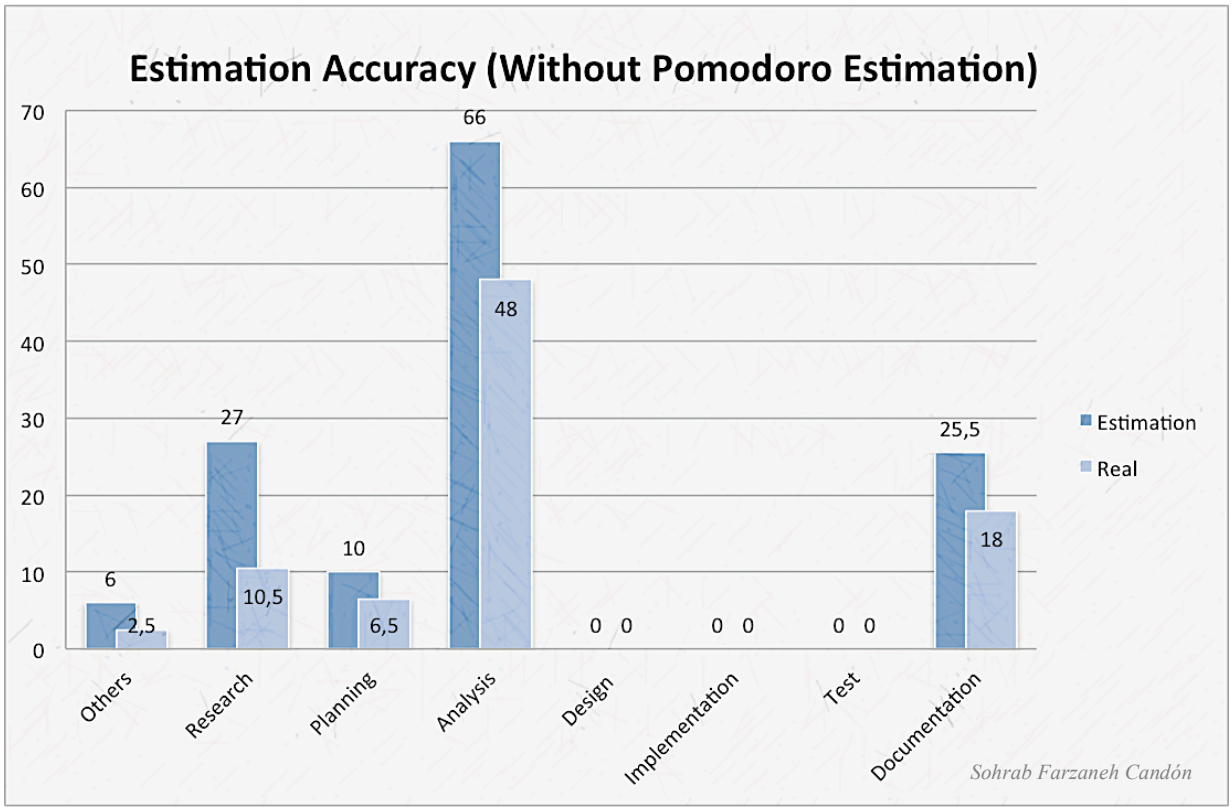

Figure 12.2-1 Estimation Accuracy (Conventional Planning)

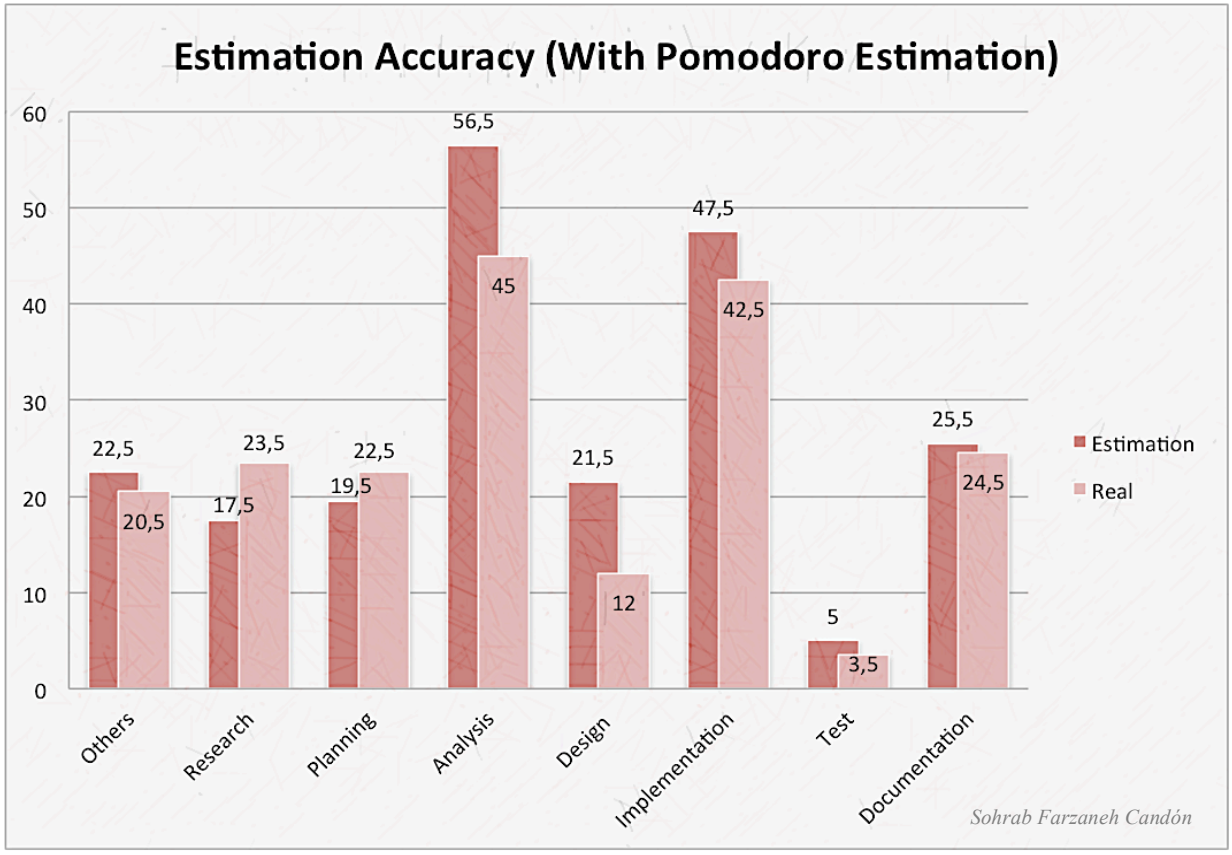

Figure 12.2-2 Estimation Accuracy (Micro-Planning)

For measuring the efficiency of the work while developing the *E-LEDA system* the accuracy between the planned time and the real time have been used, because if the planned time is accurate enough the project delivery could be specified and the resources could be reorganized to increase the productivity of all the projects carried out. The quality and maintainability of the final product are also important objectives while measuring the efficiency of the work. Since in the *E-LEDA system construction phase 1* the error correction was less than a 0,01% of the time spent in the project, it can be said that the process used to assure the quality of the final product.

The Equation 12.2-1 - Estimation Accuracy represents the operations performed in order to obtain the estimation accuracy final value. *100%* would represent a perfect estimation.

$$(1 - \frac{ABS(EsimationTime - RealTime)}{EstimationTime}) * 100$$

Equation 12.2-1 - Estimation Accuracy

Figure 12.2-3 Estimation Accuracy Comparison represents the estimation accuracy of the different techniques used while developing the *E-LEDA system* and the final estimation accuracy including both techniques. As the chart shows the accuracy grows a *18,1%* when micro planning is used. Although an important remark is that, micro-planning was used in a more advanced phase of the project, when dealing with uncertainty was not as much as in the early phases of the project, which might also influenced the improvement of the estimation accuracy.

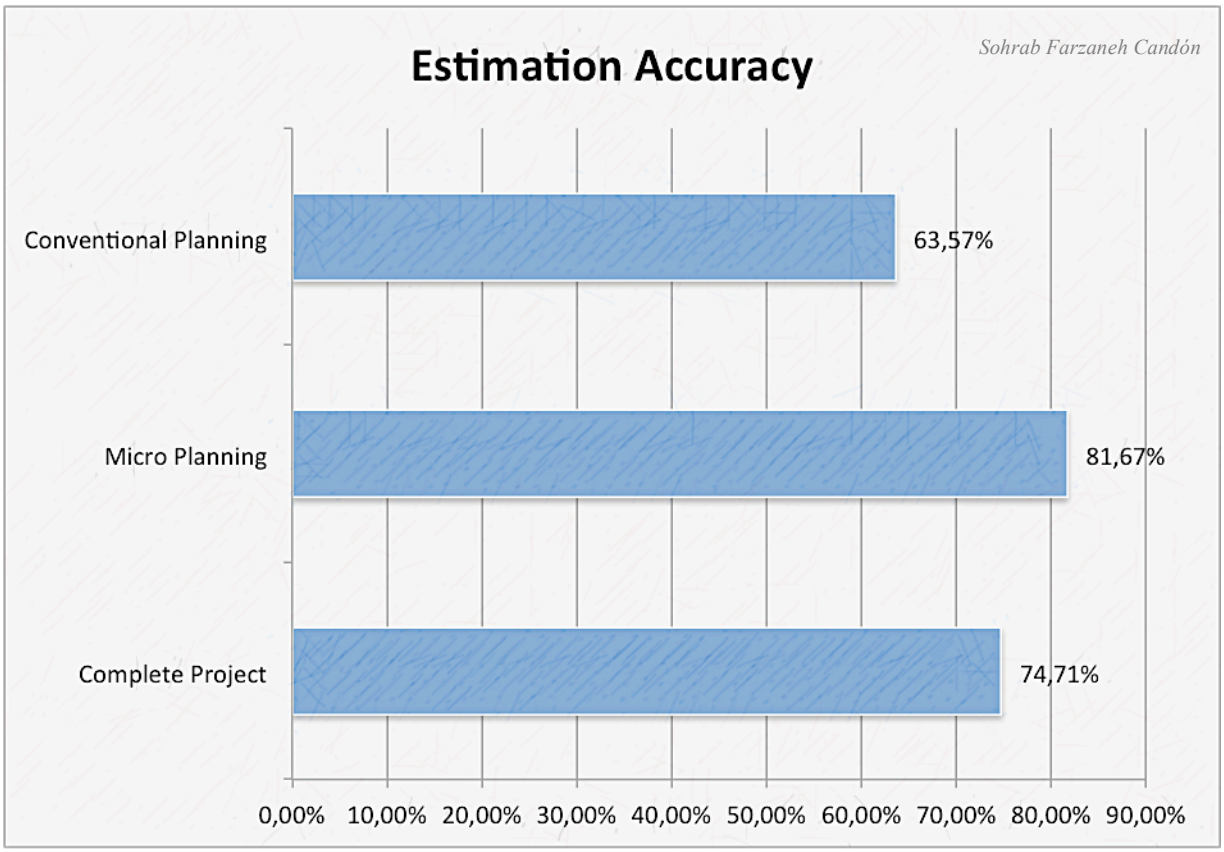

Figure 12.2-3 Estimation Accuracy Comparison

17 Future Works

As described in this document, only the first construction phase of the *E-LEDA system* has been developed, in the future the *construction phase 2 (Authentication), construction phase 3 (LMS management) and construction phase 4 (Complete iOS app)* shall be develop.

Once the complete *E-LEDA system* is developed and installed, there are plenty of features that have not been included in this version of the system, such as tablet compatibility, student access or other platforms and LMSs compatibility, some of the future requirements are described in the *specific requirements documents* as desirable future features (Farzaneh Candón, SRS, 2013).

Another wish in the future is to collaborate directly with the LMSs in order to provide a system near to their clients needs, so they can profit as much as possible from the *E-LEDA system* experience. One of the target LMS to work with would be Chamilo, since their philosophy allows the integration of new platforms and ideas.

Aside from the *E-LEDA system* a new field for research was opened with the *E-LEDA project's* working method. Since the final estimation accuracy resulted in a 18,1% improvement it might reflect that the working method could also be used for other projects, and might provide more accurate task estimation for other small software projects. Although the results are not enough to conclude the suitability of the working method for other projects a future paper is being prepared in this field.

Bibliography

Allwein, C. (2012, [s.d.] [s.d.]). *iOS Frameworks*. Retrieved April 23, 2013 from iOS Frameworks: http://iosframeworks.com/

Apple Inc. (2012, [s.d.] [s.d.]). *Developer Tools*. Retrieved October 3, 2012 from Developer: https://developer.apple.com/technologies/tools/

Apple Inc. (2010). Introducing iPhone 4. *Apple WWDC*. San José: Apple Inc.

Apple Inc. (2012). *iOS Technology Overview* (Vol. 1). Cupertino, California, USA: Apple Inc.

Apple Inc. (2007). Macworld San Francisco 2007. *Macworld conference & Expo*. San Francisco: Apple Inc.

ATutor. (2013, March 15). *ATutor: Learning Management Tools*. Retrieved April 23, 2013 from ATutor: http://atutor.ca/

Blackboard. (2013, [s.d.] [s.d.]). *Blackboard*. Retrieved April 23, 2013 from blackboard: http://www.blackboard.com/

Chamilo. (2013, April 23). *Chamilo: E-Learning & Collaboration Software*. Retrieved April 23, 2013 from Chamilo: http://www.chamilo.org/es

Cirillo, F. (2009). *The Pomodoro Technique*. [s.l]: Lulu.com.

Clark, J. (2010). *Tapworthy: Designing Great iPhone Apps*. (K. Shaner, Ed.) [s.l.], Canada: O'Reilly Media.

Core-Plot. ([s.d.], [s.d.] [s.d.]). *Core-Plot: cocoa plotting framework for OS X and iOS*. Retrieved April 23, 2013 from Google Projects: http://code.google.com/p/core-plot/

Farzaneh Candón, S. (2013). *Construction Phase 1*. Thesis, Universidad Carlos III de Madrid, Departamento de Informática, Leganés.

Farzaneh Candón, S. (2013). *E-LEDA Software Requirement Specification*. Thesis, Universidad Carlos III de Madrid, Departamento de Informática, Madrid.

Farzaneh Candón, S. (2013). *Use Case Definition*. Thesis, Universidad Carlos III de Madrid, Departamento de Ingeniería informática, Madrid.

Git. ([s.d.], [s.d.] [s.d.]). *Git*. Retrieved April 23, 2013 from Git: http://git-scm.com/

Humphrey, W. S. (2000). *Introduction to the Team Software Process* (Ilustrated, reprinted ed.). (W. S. Humphrey, Ed.) [s.l.]: Addison-Wesley Professional.

iVisualization. (2011, January 23). *Chart Library for iPhone*. Retrieved Aril 23, 2013 from iVisualization: http://ivisualization.com/

Krasner, G. E., & Pope, S. T. (1988). *A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System*. ParcPlace Systems Inc. Mountain View: ParcPlace Systems Inc.

Larman, C. (2002). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process* (2, Ilustrated ed.). (C. Larman, Ed.) [s.l]: Prentice Hall Professional.

López Hernández, F. (2008). *El lenguaje Objective-C para programadores C++ y Java* (Vol. I). Madrid, Madrid, Spain: MacProgramadores.

Mateos Carrera, C. (2011). *Definición de una plataforma tecnológica para la formación de capital humano y capitalización del conocimiento*. Thesis, Universidad Carlos III de Madrid, Departamento de Informática, Leganés.

McGalliard, J., & Agapow, P.-M. (1997). An Introduction to Objective-C. In G. Michaelson (Ed.), *AUUG'97 Conference* (pp. 236-242). Queensland: ACMS - Conventions & Exhibitions.

Microsoft. (2012, [s.d.] [s.d.]). *Model-View-Controller*. Retrieved October 3, 2012 from msdn: http://msdn.microsoft.com/en-us/library/ff649643.aspx

Microsoft. (2013, [s.d.] [s.d.]). *Office*. Retrieved April 23, 2013 from Office: http://office.microsoft.com/en-gb/

Moodle. (2013, April 17). *Moodle*. Retrieved April 23, 2013 from Moodle: https://moodle.org/

MySQL. (2013, [s.d.] [s.d.]). *MySQL*. Retrieved April 23, 2013 from MySQL: http://www.mysql.com/

NuAS. (2013, March 31). *PowerPlot - Stunning charts for the iPhone*. Retrieved April 23, 2013 from PowerPlot: http://powerplot.nua-schroers.de/

OmiGroup. (2012, February 21). *OmniPlan*. Retrieved April 23, 2013 from OmniGroup: http://www.omnigroup.com/omniplan

OmniGroup. (2012, February 21). *OmniGraffle*. Retrieved April 23, 2013 from OmniGroup: http://www.omnigroup.com/omnigraffle

Over, J. (2010). *Introduction to the Team Software Process*. Carnegie Mellon University, Software Engineering Institute. Pittsburgh: Carnegie Mellon University.

Paulsen, M. F. (2002). *Online Education Systems: Discussoin and Definition Terms*. NKI Distance Education. NKI Distance Education.

Riehle, D. (2000). *Framework Design: A Role Modelling Approach*. Ph.D. Thesis, Swiss Federal Institute of Technology, Zurich.

Ross, D. (2013, April 10). *tapkulibrary*. Retrieved April 23, 2013 from GitHub: https://github.com/devinross/tapkulibrary

s7graphview. (2012, August 07). *Verious*. Retrieved April 23, 2013 from s7graphview: http://www.verious.com/component/s7graphview/

Sakai. (2013, [s.d.] [s.d.]). *Sakai project*. Retrieved April 23, 2013 from Sakai: http://www.sakaiproject.org/

Visual Paradigm. ([s.d.], [s.d.] [s.d.]). *Visual Paradigm*. Retrieved April 23, 2013 from Visual Paradigm: http://www.visual-paradigm.com/

Vox Media Inc. (2013, February 12). *iOS A visual history*. Retrieved April 23, 2013 from The Verge: http://www.theverge.com/2011/12/13/2612736/ios-history-iphone-ipad

Glossary

TERMS

Chamilo: OpenSource e-learning and content management system

E-Learning: On-Line Learning Plattform via internet

Framework: Universal, reusable software platform

Gannt diagram: Bat chart that illustrates a project schedule

GCC: GNU compiler

IDE: *Integrated Development Environment,* environment used to help developers in a certain set of programming languages

iPad: Apple Inc. Device

iPhone: Apple Device

iPod: Apple Inc. Device

Larman's Method: Simplification of the RUP for small Object - Oriented projects

LMS(*Learning Management System*): System that organize and provides acess to on-line learning services

Model-view-controller: Design pattern that separates the user view the models and the logic of a system

Moodle: Modular Object-Oriented Dynamic Learning Environment. free source learning software platform

Multi-touch: Touch sensing surface able to recognize the presence of two or more points of contact.

Objective-C: Programming language superset of C used for iOS application development

Obsolete devices: Apple devices unable to uplgrade their operating system to the newest version (iOS 6.0)

Pomodoro Technique: Time management technique based in fixed-time divisions (pomodoros)

PSP: *Personal Software Process,* structured software process for improving the personal efficiency and productivity

RUP: *Rational Unified Process*, widely used agile methodology for software development

SDK: Software Development Kit, includes all the necesary modules to develop in a certain plattform

Smalltalk:Object oriented superset of C developed in the early1980's

UML: *Unified Modelling Language,* Standarized general purpose modelling language for object-oriented sofware projects.

XCode: Most extended IDE for Obejctive-C