myTaxiService

Project Plan

Belluschi Marco 791878, Cerri Stefano 849945, Di Febbo Francesco 852389

January 26, 2016

Contents

Chapter 1

Introduction

1.1 Purpose and scope

1.1.1 Purpose

This is the *Project Plan* (**PP**) for *myTaxiService*. Its purpose is to evaluate time and resources necessary to develop the entire *myTaxiService* system, including mobile and web applications.
To do so, the **Function Points** measurement and the **COCOMO** model have been used, comparing the results with the system size and the implementation time.

1.1.2 Scope

As more widely explained both in the RASD and in the DD, myTaxiService is a taxi service for a large city. The main goals of the system are:

- simplify the access of passengers to the service;
- guarantee a fair management of taxi queues.

See the two aforementioned documents for further explanations.

1.2 List of definitions and abbreviations

Definitions

- Function Point: unit of measurement that expresses the amount of business functionality an information system (as a product) provides to a user compared to software size; costs (in dollars or hours) are calculated from past projects

- COCOMO Model: algorithmic software cost estimation model that uses a basic regression formula with parameters derived from historical project data and current/future project characteristics
- User: person that uses the service applications
- Passenger: passenger registered to the service
- Taxi driver: taxi driver registered to the service
- Mobile App: *myTaxiService* mobile application
- Web App: *myTaxiService* web application
- System: the union of software and hardware to be developed and implemented

Acronyms

- PP: Project Plan
- RASD: Requirements Analysis and Specification Document
- DD: Design Document
- ITPD: Integration Test Plan Document
- FP: Function Point
- COCOMO: COnstructive COst MOdel
- SLOC: Source Line Of Code
- KSLOC: 1000 SLOCs
- EAF: Effort Adjustement Factor
- ILF: Internal Logic File
- EIF: External Interface File
- EI: External Input
- EO: External Output
- EQ: External Inquiry

1.3 List of reference documents

- Software Engineering 2 Project AA 2015/2016: Project Description And Rules
- Software Engineering 2 Project AA 2015/2016: Assignment 5 - Project Plan
- myTaxiService's Requirement Analysis and Specification Document (RASD)
- myTaxiService's Design Document (DD)
- myTaxiService's Integration Test Plan Document (ITPD)

Chapter 2

Function Points

2.1 Introduction

The Functional Point approach is a technique that allows to evaluate the effort needed for the design and implementation of a project. We have used this technique to evaluate the application dimension basing on the functionalities of the application itself. The functionalities list has been obtained from the RASD document and for each one of them the realization complexity has been evaluated. The functionalities has been groped in:

- Internal Logic File: it represents a set of homogeneous data handled by the system
- External Interface File: it represents a set of homogeneous data used by the application but handled by external application
- External Input: elementary operation that allows input of data in the system
- External Output: elementary operation that creates a bit stream towards the outside of the application
- External Inquiry: elementary operation that involves input and output operations

The following table outline the number of Functional Point based on functionality and relative complexity:

Function Type	**Complexity**		
	Simple	Medium	Complex
Internal Logic File	7	10	15
External Interface File	5	7	10
External Input	3	4	6
External Output	4	5	7
External Inquiry	3	4	6

2.2 FP Estimation

2.2.1 Internal Logic Files

The Internal Logic Files (ILFs) of the system are the following:

- User : medium complexity
- Passenger : simple complexity
- Taxi Driver : medium complexity
- Ride : simple complexity

2.2.2 External Interface Files

The External Interface File (EIF) of the system is the following:

- Maps : is acquired from Google Maps services. High complexity

2.2.3 External Input

The External Inputs (EIs) of the system are the following:

- User Registration : medium complexity
- User Login : simple complexity
- User Logout : simple complexity
- User Profile Management : simple complexity
- Ride Request : high complexity
- Taxi Availability : simple complexity

2.2.4 External Output

The External Outputs (EOs) of the system are the following:

- Passenger Notification : simple complexity
- Confirmation e-mails : simple complexity
- Taxi Driver Notification : simple complexity

2.2.5 External Inquiry

The External Outputs (EQs) of the system are the following:

- Ride Sharing : high complexity
- Taxi Driver ride request handling : simple complexity

2.2.6 Summary

Function Type	**Complexity-Weight**			**Total Points**
	Simple	Medium	Complex	
Internal Logic File	2×7	2×10	0×15	34
External Interface File	0×5	0×7	1×10	10
External Input	3×3	1×4	2×6	25
External Output	3×4	0×5	0×7	12
External Inquiry	1×3	0×4	1×6	9
Total Points	38	24	28	**90**

Chapter 3

COCOMO

3.1 Introduction

This chapter describes the estimation achieved through COCOMO II: a complex, non linear model that takes in account the characteristics of the product but also of people and process.

Official manual at: http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

In the previous chapter we have estimated a Function Point value. Following the table described in http://www.qsm.com/resources/function-point-languages-table we estimate a conversion factor of 46.

$$90 \text{ FPs} \times 46 = 4140 \text{ SLOC}$$

3.2 Scale Drivers

- **Precedentedness (PREC):** reflects the previous experience of the organisation with this type of project. *Very low* means no previous experience, *Extra high* means that the organisation is completely familiar with this application domain.
- **Development flexibility (FLEX):** reflects the degree of flexibility in the development process. *Very low* means a prescribed process is used; *Extra high* means that the client only sets general goals.
- **Architecture/Risk resolution (RESL):** reflects the extent of risk analysis carried out. *Very low* means little analysis, *Extra high* means a complete thorough risk analysis.
- **Team cohesion (TEAM):** reflects how well the development team know each other and work together. *Very low* means very difficult interactions,

Extra high means an integrated and effective team with no communication problems.

- **Process Maturity (PMAT):** reflects the process maturity of the organisation.

Scale driver	Factor	Value
PREC		
FLEX		
RESL		
TEAM		
PMAT		
Total points		

3.3 Cost Drivers

- **Required Software Reliability (RELY):** this is the measure of the extent to which the software must perform its intended function over a period of time. If the effect of a software failure is only slight inconvenience then RELY is very low. If a failure would risk human life then RELY is very high.

- **Data Base Size (DATA):** this cost driver attempts to capture the effect large test data requirements have on product development. The rating is determined by calculating D/P, the ratio of bytes in the testing database to SLOC in the program. The reason the size of the database is important to consider is because of the effort required to generate the test data that will be used to exercise the program. In other words, DATA is capturing the effort needed to assemble and maintain the data required to complete test of the program.

- **Product Complexity (CPLX):** it is divided into five areas: control operations, computational operations, device-dependent operations, data management operations, and user interface management operations. Each area or combination of areas characterize the product or the component of the product to be developed. The complexity rating is the subjective weighted average of the selected area ratings.

- **Developed for Reusability (RUSE):** this cost driver accounts for the additional effort needed to construct components intended for reuse on current or future projects. This effort is consumed with creating more generic design of software, more elaborate documentation, and more extensive testing to ensure components are ready for use in other applications. Development for reusability imposes constraints on the project's

RELY and DOCU ratings. The RELY rating should be at most one level below the RUSE rating. The DOCU rating should be at least Nominal for Nominal and High RUSE ratings, and at least High for Very High and Extra High RUSE ratings.

- **Documentation Match to Life-Cycle Needs (DOCU):** several software cost models have a cost driver for the level of required documentation. The rating scale for the DOCU cost driver is evaluated in terms of the suitability of the projects documentation to its life-cycle needs. The rating scale goes from Very Low (many life-cycle needs uncovered) to Very High (very excessive for life-cycle needs). Attempting to save costs via Very Low or Low documentation levels will generally incur extra costs during the maintenance portion of the life-cycle. Poor or missing documentation will increase the Software Understanding.

- **Execution Time Constraint (TIME):** this is a measure of the execution time constraint imposed upon a software system. The rating is expressed in terms of the percentage of available execution time expected to be used by the system or subsystem consuming the execution time resource. The rating ranges from nominal, less than 50% of the execution time resource used, to extra high, 95% of the execution time resource is consumed.

- **Main Storage Constraint (STOR):** this rating represents the degree of main storage constraint imposed on a software system or subsystem. The rating ranges from nominal (less than 50%), to extra high (95%).

- **Platform Volatility (PVOL):** Platform is used here to mean the complex of hardware and software (OS, DBMS, etc.) the software product calls on to perform its tasks. If the software to be developed is an operating system then the platform is the computer hardware. If a database management system is to be developed then the platform is the hardware and the operating system. If a network text browser is to be developed then the platform is the network, computer hardware, the operating system, and the distributed information repositories. The platform includes any compilers or assemblers supporting the development of the software system. This rating ranges from low, where there is a major change every 12 months, to very high, where there is a major change every two weeks.

- **Analyst Capability (ACAP):** Analysts are personnel who work on requirements, high-level design and detailed design. The major attributes that should be considered in this rating are analysis and design ability, efficiency and thoroughness, and the ability to communicate and cooperate. The rating should not consider the level of experience of the analyst; that is rated with APEX, LTEX, and PLEX. Analyst teams that fall in the fifteenth percentile are rated very low and those that fall in the ninetieth percentile are rated as very high, see Table 26.

- **Programmer Capability (PCAP):** Current trends continue to emphasize the importance of highly capable analysts. However the increasing role of complex COTS packages, and the significant productivity leverage associated with programmers ability to deal with these COTS packages, indicates a trend toward higher importance of programmer capability as well. Evaluation should be based on the capability of the programmers as a team rather than as individuals. Major factors which should be considered in the rating are ability, efficiency and thoroughness, and the ability to communicate and cooperate. The experience of the programmer should not be considered here; it is rated with APEX, LTEX, and PLEX. A very low rated programmer team is in the fifteenth percentile and a very high rated programmer team is in the ninetieth percentile, see Table 27.

- **Personnel Continuity (PCON):** The rating scale for PCON is in terms of the projects annual personnel turnover: from 3

- **Applications Experience (APEX):** The rating for this cost driver (formerly labeled AEXP) is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project teams equivalent level of experience with this type of application. A very low rating is for application experience of less than 2 months. A very high rating is for experience of 6 years or more, see Table 29.

- **Platform Experience (PLEX):** The Post-Architecture model broadens the productivity influence of platform experience, PLEX (formerly labeled PEXP), by recognizing the importance of understanding the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities, see Table 30.

- **Language and Tool Experience (LTEX):** This is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem. Software development includes the use of tools that perform requirements and design representation and analysis, configuration management, document extraction, library management, program style and formatting, consistency checking, planning and control, etc. In addition to experience in the projects programming language, experience on the projects supporting tool set also affects development effort. A low rating is given for experience of less than 2 months. A very high rating is given for experience of 6 or more years, see Table 31.

- **Use of Software Tools (TOOL):** Software tools have improved significantly since the 1970s projects used to calibrate the 1981 version of COCOMO. The tool rating ranges from simple edit and code, very low, to integrated life-cycle management tools, very high. A Nominal TOOL rating in COCOMO 81 is equivalent to a Very Low TOOL rating in COCOMO II. An emerging extension of COCOMO II is in the process of

elaborating the TOOL rating scale and breaking out the effects of TOOL capability, maturity, and integration, see Table 32.

- **Multisite Development (SITE):** Given the increasing frequency of multisite developments, and indications that multisite development effects are significant, the SITE cost driver has been added in COCOMO II. Determining its cost driver rating involves the assessment and judgement-based averaging of two factors: site collocation (from fully collocated to international distribution) and communication support (from surface mail and some phone access to full interactive multimedia). For example, if a team is fully collocated, it doesnt need interactive multimedia to achieve an Extra High rating. Narrowband e-mail would usually be sufficient, see Table 33.

- **Required Development Schedule (SCED):** This rating measures the schedule constraint imposed on the project team developing the software. The ratings are defined in terms of the percentage of schedule stretch-out or acceleration with respect to a nominal schedule for a project requiring a given amount of effort. Accelerated schedules tend to produce more effort in the earlier phases to eliminate risks and refine the architecture, more effort in the later phases to accomplish more testing and documentation in parallel. In Table 34, schedule compression of 75schedule stretch-out of 160Their savings because of smaller team size are generally balanced by the need to carry project administrative functions over a longer period of time. The nature of this balance is undergoing further research in concert with our emerging CORADMO extension to address rapid application development (goto http://sunset.usc.edu/COCOMOII/suite.html for more information). SCED is the only cost driver that is used to describe the effect of schedule compression / expansion for the whole project. The scale factors are also used to describe the whole project. All of the other cost drivers are used to describe each module in a multiple module project. Using the COCOMO II Post-Architecture model for multiple module estimation is explained in Section 3.3.

Cost driver	Factor	Value
RELY		
DATA		
CPLX		
RUSE		
DOCU		
TIME		
STOR		
PVOL		
ACAP		
PCAP		
PCON		
APEX		
PLEX		
LTEX		
TOOL		
SITE		
SCED		
Total points		

3.4 Schedule Estimation

Chapter 4

Schedule

Chapter 5

Resources

Chapter 6

Risks

Appendix A

Appendix

A.1 Software and tool used

A.2 Working hours