Project Plan

02/02/2016

Elis Bardhi **790135**

Andrea Cavalli **841512**

Mario Dolci **773705**

1. Introduction

In this document the time and resources necessary to MyTaxiService project are evaluated. There is also a description of the assignments that have been done during this project, then an estimation of the risks linked to MyTaxiService are explained, with their probabilities, impacts and solutions.

2. Function Point and COCOMO Approach

2.1 Counting with Function Points FP

The Function Point estimation approach is based on the amount of functionalities in a software and their complexity. Function Points estimators are useful because they are based on information that is available early in the project life cycle.

To perform this estimation, we have based our parameters on the following tables, taken from COCOMO II, Model Definition Manual at:

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

- This first schema is used for determining complexity-level function counts. It allows us to classify each functionality into Low, Average and High complexity levels.

Table 2. FP Counting Weights

For Internal Logical Files and External Interface Files

Record Elements	Data Elements 1 - 19	Data Elements 20 - 50	Data Elements 51+
1	Low	Low	Avg.
2 - 5	Low	Avg.	High
6+	Avg.	High	High

For External Output and External Inquiry

File Types	Data Elements 1 - 5	Data Elements 6 - 19	Data Elements 20+
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
4+	Avg.	High	High

For External Input

File Types	Data Elements 1 - 4	Data Elements 5 - 15	Data Elements 16+
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
3+	Avg.	High	High

- The following one defines the weights values that we've to use to perform the FP value.

Table 3. UFP Complexity Weights

Function Type	Complexity-Weight Low	Complexity-Weight Average	Complexity-Weight High
Internal Logical Files	7	10	15
External Interfaces Files	5	7	10
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6

- Then we need to convert the FP to lines of code. We will be using the table found at this URL: http://www.qsm.com/resources/function-point-languages-table

[SLOC/UFP] 53

2.2 FP Estimation

2.2.1. Internal Logical Files

The application includes a number of ILFs that will be used to store the information about clients, taxi drivers, taxis, taxi requests, route, city zones and GPS location. Now we analyze this section more in detail.

The system stores few information about clients and taxi drivers such as name, surname, username, password and email. All this are simple strings, integers and data-time, so we will classify them with a medium complexity.

Like above, taxies are described with simple information such as plate, frame number and production year, so these data will have a medium complexity.

Then, for what concern a taxi request, it will be classified with a high complexity because it contains complex data structures that are used to manage taxi requests or reservations.

Finally, routes, city zones and GPS location will be classified respectively with a medium, simple and medium complexity.

ILF	Complexity	FP
Client	Medium	10
Taxi driver	Medium	10
Taxi	Medium	10
Request	High	15
Route	Medium	10
City zone	Simple	7
GPS location	Medium	10
Total		**72**

2.2.2. External Logical Files

Since the main system uses an external service for tracking the taxis, we have also to include this in our cost estimation. In the RASD document we have assumed that data incoming from the external service are encoded in a JSON structure, so we can classify this with a high complexity since JSON is not a simple structure.

ELF	Complexity	FP
GPS tracking	Complex	10
Total		**10**

2.2.3. External Inputs

The application interacts with clients and taxi drivers and allows them to:

- Client
  - *Login /Logout* : simple operations
  - *Modify credentials* : medium operation
  - *Ask for a taxi (normal or reservation)* : this is a complex operation since it involves many operations such as a route calculation
  - *Manage requests (modify or delete)* : this can be classified with a high complexity since it involves many operations
  - *Manage taxi sharing* : this is a medium operation
- Taxi driver
  - *Login /Logout* : these are simple operations
  - *Modify credentials* : medium operation
  - *Manage taxi status* : medium operation
  - *Manage client requests* : complex operation because involves *many operations*

The results are resumed in the following table.

EI	Complexity	FP
Login /Logout	Simple	3x4
Modify credentials	Medium	4x2
Ask for taxi	High	6
Manage request	High	6
Manage taxi sharing	Medium	4
Manage taxi status	Simple	4
Manage client requests	High	6
Total		**46**

2.2.4. External Outputs

The application performs two external outputs:

- the suggestions sent to users while they are compiling the string form to insert street names and numbers;
- notifications sent to users to inform them about their requests or if something is changed.

Both this has a medium complexity.

EO	Complexity	FP
Suggestions	Medium	5
Notifications	Medium	5
Total		**10**

2.2.5. External Inquiries

The application allows taxi drivers to request information about theri clients to request information about:

- their profiles;
- rides that they have asked for;
- the route of a ride;
- ride information, such as number of passengers or direction;

and allow taxi drivers to request information's about their and clients profiles.

EQ	Complexity	FP
Visualize profile	Simple	3x3
Ride information's	Complex	6
Visualize rides chronology	Medium	4
Visualize route	Complex	6
Total		**22**

2.2.6. Resuming

By summing up all these numerical values, we get a total estimation of 160 FPs. This value is used to get the estimation of the number of lines of code. By using the parameter that was wrote in the first paragraph, we get a SLOC equal to **8480**.

The following table resumes our estimations:

Function type	Value
Internal Logical File	72
External Logical File	10
External Input	46
External Output	10
External Inquiries	22
Total	**160**

2.3 Cocomo II

2.3.1. Brief Introduction

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

All the tables used in this analysis have been taken from COCOMO II, Model Definition
http://csse.usc.edu/tools/COCOMOII.php

2.3.2. Scale Drivers

Table 10. Scale Factor Values, SF_j, for COCOMO II Models

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unpreceden ted	largely unpreceden ted	somewhat unpreceden ted	generally familiar	largely familiar	thoroughly familiar
SF_j:	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_j:	5.07	4.05	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_j:	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_j:	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	The estimated Equivalent Process Maturity Level (EPML) or					
	SW-CMM Level 1 Lower	SW-CMM Level 1 Upper	SW-CMM Level 2	SW-CMM Level 3	SW-CMM Level 4	SW-CMM Level 5
SF_j:	7.80	6.24	4.68	3.12	1.56	0.00

These values are evaluated according to the following table:

- ***Precedentedness**:* it reflects the previous experience that we had with this kind of projects. Since for us this was the first experience using this framework and these developments methodologies, this value will be low.
- ***Development flexibility**:* it reflects the degree of flexibility in the development process. The professors set the general specifications without going too much in detail, for this reason this value will be high.
- ***Risk resolution**:* it reflects the extent of risk analysis carried out. Thanks to filters, and security access mostly of the risks were eliminated then this value will be very high.
- ***Team cohesion**:* it reflects how well the development team know each other and work together. In our case we had some problems, in particular for the difference of working time

and initial synchronization issues. Nonetheless, we overcame those difficulties by thoroughly describing guidelines and goals in our development process. Since this approach was successful the final value for this attribute is very high.

- ***Process maturity:*** this was evaluated around the 18 Key Process Area (KPAs) in the SEI Capability Model. Because of the goals were consistently achieved these values will be set to high, level 3.

The results are resumed in the following table.

Scale Driver	Factor	Value
Precedentedness	Low	4.96
Development Flexibility	High	2.03
Risk Resolution	Very High	1.41
Team Cohesion	Very High	2.19
Process Maturity	High	3.12
Total		**13.71**

2.3.2. Cost Drivers

Since we haven't implemented the system some values are invented.

- **Required Software Reliability (RELY):**

  This measure is set to high, in particular because software failures may cause financial losses of money.

Table 17. RELY Cost Driver

RELY Descriptors:	slight inconven-ience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.82	0.92	1.00	1.10	1.26	n/a

- **Database Size:**

  It translates the effects that large data have in our application. Our test database size is equal to 1000 KB and the program size is equal to **8480** SLOC, the division D/P = 117.92 and then this parameter has a high value 1.14.

Table 18. DATA Cost Driver

DATA* Descriptors		Testing DB bytes/Pgm SLOC < 10	10 ≤ D/P < 100	100 ≤ D/P < 1000	D/P ≥ 1000	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.90	1.00	1.14	1.28	n/a

- **Product Complexity:**

  It is set to Very High according to the new COCOMO II CPLEX rating scale.

Table 20. CPLX Cost Driver

Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.73	0.87	1.00	1.17	1.34	1.74

- **Required Reusability (RUSE):**

  In our project there are different reusable components since our aim was to design the system as modular as possible. Since the RELY should be at most one level below the RUSE, we set this to high.

Table 21. RUSE Cost Driver

RUSE Descriptors:		none	across project	across program	across product line	across multiple product lines
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.95	1.00	1.07	1.15	1.24

- **Documentation match to life-cycle needs:**

  This parameter describes the relation between the provided documentation and the application requirements. Its suitability is set to nominal since each aspect of our system to be described has been expressed in the RASD or in the DD. On the other hand, there is no part of those document unrelated to the actual phase of the development the document is addressed to.

Table 22. DOCU Cost Driver

DOCU Descriptors:	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.81	0.91	1.00	1.11	1.23	n/a

- **Execution Time Constraint:**

  In our case this parameter is not relevant, so is reasonable to set it as very low.

Table 23. TIME Cost Driver

TIME Descriptors:			≤ 50% use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.11	1.29	1.63

- **Main Storage Constraint:**

  This parameter represents the degree of main storage constraint. In our application this parameter is not relevant so is set as very low.

Table 24. STOR Cost Driver

STOR Descriptors:			≤ 50% use of available storage	70% use of available storage	85% use of available storage	95% use of available storage
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.05	1.17	1.46

- **Platform Volatility:**

  In our application is reasonable to consider as platforms the DBMS, the operating system, the browser that perform injections and the hardware as far as the environment concerns. We have to consider also the compiler and the web server that has taken an important role in the developing phase. The platform should not change too often, so this value is set to low.

Table 25. PVOL Cost Driver

PVOL Descriptors:		Major change every 12 mo.; Minor change every 1 mo.	Major: 6 mo.; Minor: 2 wk.	Major: 2 mo.;Minor: 1 wk.	Major: 2 wk.;Minor: 2 days	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.87	1.00	1.15	1.30	n/a

- **Analyst Capability:**

  Design and analysis abilities should be set to high, since we intentionally dedicated a lot of effort in analyzing the problem requirements and its potential integration in a real word scenario. In particular, not only we can grant that the requirements have been correctly studied and accomplished, but also that our design makes our application actually useful for an end user, providing each of the basic functionalities he may need. In particular, we resolved any ambiguity present in the initial description and explained our solution in the RASD.

Table 26. ACAP Cost Driver

ACAP Descriptors:	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.42	1.19	1.00	0.85	0.71	n/a

- **Programmer Capability:**

  This parameter is evaluated according to our degree of cooperation and, due to some small problems on it, it is set to high.

Table 27. PCAP Cost Driver

PCAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.34	1.15	1.00	0.88	0.76	n/a

- **Application Experience:**

  Our project experience is evaluated according to our previous experience in web projects and also according to our abilities in programming in PHP, Android and Java, so this parameter is set to high.

Table 29. APEX Cost Driver

APEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 years	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.10	1.00	0.88	0.81	n/a

- **Platform Experience:**

  Our average knowledge about platforms such as databases, user interfaces and server side development are around 3 years, so this parameter is set to high.

Table 30. PLEX Cost Driver

PLEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.19	1.09	1.00	0.91	0.85	n/a

- **Language and Tool Experience:**

  This parameter reflects the same experience of the previous one, so it is set to high.

Table 31. LTEX Cost Driver

LTEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.20	1.09	1.00	0.91	0.84	

- **Personnel continuity:**

  This parameter is relevant in particular since in the current case our available time is less than half a year. For this reason, we set it to very low.

Table 28. PCON Cost Driver

PCON Descriptors:	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.29	1.12	1.00	0.90	0.81	

- **Usage of Software Tools:**

  We used NetBeans and Intellij IDEA with Maven to manage dependencies of our project such as libraries and development kits, and Git for the repository management. The most appropriate value is nominal.

Table 32. TOOL Cost Driver

TOOL Descriptors	edit, code, debug	simple, frontend, backend CASE, little integration	basic life-cycle tools, moderately integrated	strong, mature life-cycle tools, moderately integrated	strong, mature, proactive life-cycle tools, well integrated with processes, methods, reuse	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.17	1.09	1.00	0.90	0.78	n/a

- **Multisite development:**

  This parameter reflects how we handled the distribution of development over distance and multiple platforms. We have used phones, mail and Skype also with screen sharing, so this value is set to extra high.

Table 33. SITE Cost Driver

SITE: Collocation Descriptors:	Inter-national	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro. area	Same building or complex	Fully collocated
SITE: Communications Descriptors:	Some phone, mail	Individual phone, FAX	Narrow band email	Wideband electronic communication.	Wideband elect. comm., occasional video conf.	Interactive multimedia
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.09	1.00	0.93	0.86	0.80

- **Required development schedule:**

  Our efforts were well distributed over the available development time, but regardless of this fact, the implementation required high efforts at the later phases. Mainly this is due to the fact that we expanded the initial problem description in the more complex and profitable way for a real world application. For these reason this parameter should be set to high.

Table 34. SCED Cost Driver

SCED Descriptors	75% of nominal	85% of nominal	100% of nominal	130% of nominal	160% of nominal	
Rating Level	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multiplier	1.43	1.14	1.00	1.00	1.00	n/a

Our results are expressed in the following table.

Scale Driver	Factor	Value
Required Software Reliability	High	1.10
Data Base Size	High	1.14
Product Complexity	High	1.34
Required Reusability	High	1.07
Documentation match to life-cycle needs	Nominal	1.00
Execution Time Constraint	Very Low	n\a
Main Storage Constraint	Very Low	n\a
Platform Volatility	Low	0.87
Analyst Capability	High	0.85
Programmer Capability	High	0.88
Application Experience	High	0.88
Platform Experience	High	0.91
Language and Tool Experience	High	0.91
Personnel continuity:	Very Low	1.29
Usage of Software Tools	Nominal	1.00
Multisite development	Extra High	0.80
Required development schedule	High	1.00
Product		**0.88**

2.3.3. Effort Equation

This final equation gives us the effort estimation measured in Person-Months (PM):

$$Effort = A * EAF * KSLOC^{E}$$

Where:

- A → 2.94 (for COCOMO.2000)
- EAF → product of all the cost drivers, equal to: 0.88
- E → exponent derived from Scale Drivers. Is calculated as:
    - B + 0.01 * sum{i}
    - SF[i]:= B + 0.01 *15.04 = 0.91 + 0.1504 = 1.0604
      in which B is equal to: 0.91 for COCOMO.2000.
- KSLOC → estimated lines of code using the FP analysis: **8.480**

With these parameters we can compute the Effort value, that is equal to:

$$Effort = 2{,}94 * 0.88 * 8.480^{1{,}0604} = 24.96\ PM$$

Since we have not developed the program we use the estimated KLOC (**8.480**)

2.3.4. Duration Equation

As far as the schedule estimation, we are going to use the following formula.

$$Duration = 3{,}67 * Effort^{F}$$

$$\mathrm{F} = 0.28 + 0.2 * (1{,}0604 – 0{,}91) = 0.31$$

$$\mathrm{Duration} = 3.67\ *\ 24{,}96^{0{,}31} = 9.949$$

The duration calculated here is not even close to the actual time that we had to our disposal for this project, that could be estimated being around 4 months.

As for the required number of people the estimation is:

P = Effort / Duration = 24.96/9.949 = 2.508 ~ 3

This result, since we are actually a team of tree persons, is coherent with the reality of the development environment keeping present that KSLOC is not real, it is estimated.

3. Tasks and Schedule

The project is composed of five assignments:

- **RASD Document**: it contains the description of actors and the scenarios of MyTaxiService, the use cases that describe them, and models describing requirements and specification for the system. In this document a large use of UML is provided and the Alloy tool is used for the analysis of MyTaxiService.
- **Design Document**: it contains a functional description of the system. In this document the components of MyTaxiService are described and the relationships between them is analyzed. The document provides also a description of the main algorithms (calculation of the best route, calculation of the ride cost and the management of the zone queues).
- **Code inspection**: in this assignment a systematic examination of a computer source code has been done. We have analyzed a JEE class and four methods in it searching for coding errors or possible improvements.
- **ITPD Document**: this document the decisions for testing the integration of MyTaxiService components and its purpose is to test the interfaces between between the components that have already described in the Design Document;
- **Project Plan Document**: in this document a cost estimation of MyTaxiService system is calculated using Function Points and COCOMO; it also presents an overall description of the project' tasks, the members of the group that have worked on and a list of the risks for the project.

Each assignment has got the following submission deadlines:

- RASD assignment: 6/11/2015
- DD assignment: 4/12/2015
- Code Inspection assignment: 05/01/2016
- ITPD assignment: 21/01/2016
- Project Plan assignment: 02/02/2016

4. Resources for the tasks

Here are listed the real hours of work spent for MyTaxiService project.

- RASD assignment:
  - Elis Bardhi: 29 hours;
  - Andrea Cavalli: 25 hours;
  - Mario Dolci: 24 hours;
- DD assignment:
  - Elis Bardhi: 20 hours;
  - Andrea Cavalli: 19 hours;
  - Mario Dolci: 12 hours;
- Code inspection assignment:
  - Elis Bardhi: 6 hours;
  - Andrea Cavalli: 5 hours;
  - Mario Dolci: 5 hours;
- ITPD assignment:
  - Elis Bardhi: 6 hours;
  - Andrea Cavalli: 5 hours;
  - Mario Dolci: 5 hours;
- Project Plan assignment:
  - Elis Bardhi: 6 hours;
  - Andrea Cavalli: 5 hours;
  - Mario Dolci: 3 hours.

The total hours of work spent during all phases of the project are 175 hours.

175 hours / (40*4) hours = 1.094 Person/Month

where 40 hours are the amount of time that a man can work in a week.

Here is a comparative table between the estimated value and the actual value.

	Estimated Value	Effective Value
Effort	24.96 person/months	1.094 person/months
Duration	9.949 months	4 months
Number of People	2.508	3

The estimation of COCOMO II is oversized respect to the real time spent for the project: this is due to the statistic nature of COCOMO II.

5. Risks and their relevance

Performing a risk assessment is an important step in being prepared for possible problems that can occur with MyTaxiService project. During this phase, if a potential risk is identified, a solution or plan of action should be developed.
For each risk, it is important to estimate its probability that it will occur and its impact (negligible, marginal, critical, catastrophic).

5.1. Risks Identification

In this phase risks are identified and their probability and rank are estimated.

5.1.1. Schedule risks

Risk	Probability	Impact	Solution
Schedule is optimistic (only best cases and expected cases are described)	High	Critical	Schedule must consider all possible scenarios
Schedule is based only on the use of specific team members, but them are not available	Low	Catastrophic	Schedule must be as portable as possible and its modularity has to be high
Development product is larger than estimated	Medium	Low	Development estimation must be as accurate as possible
Excessive schedule pressure reduces productivity	Low	Critical	Schedule work has to be intelligently distributed among actions
Unfamiliar theme or the product take more time than expected to design and implement	High	Critical	The product must be descripted as best as possible

5.1.2. Organization and Management Risks

Risk	Probability	Impact	Solution
Nontechnical third-part tasks take longer than expected (legal controls, budget approvals)	Medium	Marginal	Bureaucracy stuff must be done as soon as possible
Management makes decisions that reduce the development team's motivation	Low	Critical	Decisions must be taken in a democratic way
Management decisions cycle is slower than expected	Medium	Critical	Management decision cycle estimation must be as accurate as possible
Management or marketing insists on technical decision that lengthen the schedule	Low	Critical	Management decision must be as accurate as possible

5.1.3. Development Environment Risks

Risk	Probability	Impact	Solution
Development tools do not provide the planned productivity	Medium	Critical	The choses of the development tools to use must be as accurate as possible
Error-prone modules requires more testing and implementation work than expected	High	Catastrophic	The decisions in the development phase must be as accurate as possible
Requirement to operate under multiple operating systems and devices takes longer to satisfy requests than expected	High	Critical	More development teams could work in parallel on different operating systems
Working on an unfamiliar software environment causes unforeseen problems	High	Critical	Software environment training could be done

5.1.4. Personnel Risks

Risk	Probability	Impact	Solution
Personnel need extra time to learn unfamiliar software tools	Medium	Marginal	Software tools training could be done
Personnel need extra time to learn a new programming language	High	Critical	Software environment training could be done
Development team members leave before the project is complete	Low	Catastrophic	The number of members in the development team must be calculated as accurate as possible for every scenario
New development team members are added later in the project and they need to be trained	Medium	Critical	A tutor can help the new member working on the project for a while
Not enough personnel is available for the project	High	Catastrophic	The number of members in the development team must be calculated as accurate as possible for every scenario

5.1.5. Design and Implementation Risks

Risk	Probability	Impact	Solution
Overly simple design fails to address major issues and leads to redesign an reimplementation	Medium	Catastrophic	The decisions in the development phase must be as accurate as possible
Components developed separately cannot be integrated easily and they require redesign and rework	Low	Catastrophic	The decisions in the development phase must be as accurate as possible