

Project Plan Document

Version 1.0

Alberto Mario Pirovano

Alessandro Vetere

24/01/2016



POLITECNICO
MILANO 1863

Contents

1	Introduction	2
1.1	Purpose and Scope	2
1.2	List of Definitions and Abbreviations	2
1.3	List of Reference Documents	2
2	Function Points and COCOMO	3
2.1	Function Points	3
2.1.1	Internal Logic File	4
2.1.2	External Interface File	5
2.1.3	External Input	6
2.1.4	External Output	7
2.1.5	External Inquiry	8
2.1.6	Summary	9
2.2	COCOMO	9
3	Project Tasks and Scheduling	13
3.1	Project Tasks	13
3.2	Tasks Scheduling	13
3.2.1	Tasks, Durations and Dependencies	13
3.2.2	Gantt Diagram	15
4	Resources Allocation	16
4.1	Staff Allocation Chart	16
5	Project Risks	17
5.1	Risk Types	17
6	Appendix	20
6.1	Tools	20
6.2	Hours of Work	20
6.3	Revision History	20

1 Introduction

This section contains a brief introduction to the **Project Plan Document**.

1.1 Purpose and Scope

The purpose of this document is to explain the Project Plan devised for the system to be. All the people involved in the project could be considered as possible readers of the document, but the document itself is more of a guide for the Project Manager and the Management in general. The Project Plan consists in tables, Gantt diagrams, charts and natural language descriptions of the planning, scheduling and management of *myTaxiService* development.

1.2 List of Definitions and Abbreviations

In the document are often used some technical terms whose definitions are here reported:

- See the correspondent section in the **RASD** for more definitions.

For sake of brevity, some acronyms and abbreviations are used:

- **PPD**: Project Plan Document.
- **FP**: Function Points.
- **ILF**: Internal Logic File.
- **ELF**: External Logic File.
- **EI**: External Input.
- **EO**: External Output.
- **EIQ**: External Inquiry.
- **COCOMO**: Constructive Cost Model.
- See the correspondent section in the **RASD** for more acronyms and abbreviations.

1.3 List of Reference Documents

- *myTaxiService* **RASD v1.3**: Requirements Analysis and Specification Document
- **Assignment 5 - Project Plan**: Project Plan specification document
- **COCOMO II - Model Definition Manual**: [Document link](#)

2 Function Points and COCOMO

2.1 Function Points

Under the assumption that the dimension of the software can be characterized by correlating the kind of functionalities offered with the source lines of code (SLOC) of the software itself the Function Points approach, defined in 1975 by Allan Albrecht, consist in a technique to assess the effort needed to design and develop custom software applications. This technique consists in combining the following program characteristics to obtain a final result:

- External inputs and outputs,
- User interactions,
- External Interfaces,
- Files used by the system.

A weight is associated to each of these characteristics and then final calculations are done in order to obtain an indicative size of the software project in terms of SLOC.

The following table relates each types of functionality offered by the system with a weight that depends on the level of complexity of the functionality itself:

Function Types	Simple Weight	Medium Weight	Complex Weight
Internal Logic File	7	10	15
External Logic File	5	7	10
External Input	3	4	6
External Output	4	5	7
External Inquiry	3	4	6

These weights are then referred to as $w_{i,j}$ in the following sections, where i stands for the level of complexity, and j for the function type. After this section comes the in-details description of each software function class, with the relevant partial sum indicated as FP_i , where i is the abbreviation for the considered function class.

2.1.1 Internal Logic File

Internal Logic Files are homogeneous sets of data used and managed by the application.

Summary of the calculated weights:

$$\begin{aligned}FP_{ILF} &= 9 \cdot w_{Simple,ILF} + 1 \cdot w_{Medium,ILF} \\ &= 9 \cdot 7 + 1 \cdot 10 \\ &= 73\end{aligned}$$

In details:

- **Registered Passenger:**

Personal Registered Passenger data. Infrequent modifications, but frequent insertions and readings.

$$1 \cdot w_{Simple,ILF}$$

- **Taxi Driver:**

Taxi Driver data. Very infrequent modifications, infrequent insertions, frequent readings.

$$1 \cdot w_{Simple,ILF}$$

- **Administrator:**

Administrator data. Very infrequent modifications, insertions and readings.

$$1 \cdot w_{Simple,ILF}$$

- **Taxi Car:**

Taxi Car data. Very infrequent modifications, insertions and readings.

$$1 \cdot w_{Simple,ILF}$$

- **Taxi Request:**

Taxi Request data. Frequent insertions and readings, infrequent modifications.

$$1 \cdot w_{Simple,ILF}$$

- **Taxi Reservation:**

Taxi Reservation data. Frequent insertions and readings, infrequent modifications.

$$1 \cdot w_{Simple,ILF}$$

- **Problem:**

Problem data. Infrequent insertions, readings and modifications.

$$1 \cdot w_{Simple,ILF}$$

- **Zone:**

Zone data. Very infrequent insertions, readings and modifications.

$$1 \cdot w_{Simple,ILF}$$

- **GPS Data:**

GPS Data, Database Server side. Very frequent insertions and readings, but very infrequent modifications.

$$1 \cdot w_{Medium,ILF}$$

- **Taxi Drivers Queue:**

Taxi Drivers Queue data. Frequent insertions, readings and modifications.

$$1 \cdot w_{Simple,ILF}$$

2.1.2 External Interface File

External Interface Files are homogeneous sets of data used by the application but generated and maintained by other applications.

Summary of the calculated weights:

$$\begin{aligned} FP_{ELF} &= 2 \cdot w_{Simple,ELF} + 1 \cdot w_{Medium,ELF} \\ &= 2 \cdot 5 + 1 \cdot 7 \\ &= 17 \end{aligned}$$

In details:

- **Google Maps API:**

Google Maps API related data about Travel Time and Addresses. Very frequent readings.

$$2 \cdot w_{Simple,ELF}$$

- **GPS Data:**

GPS Data, Taxi Driver side. Very frequent insertions and readings, but very infrequent modifications.

$$1 \cdot w_{Medium,ELF}$$

2.1.3 External Input

External Inputs are elementary operations to elaborate data coming from the external environment.

Summary of the calculated weights:

$$\begin{aligned} FP_{EI} &= 4 \cdot w_{Simple,EI} + 3 \cdot w_{Medium,EI} + 1 \cdot w_{Complex,EI} \\ &= 4 \cdot 3 + 3 \cdot 4 + 1 \cdot 6 \\ &= 30 \end{aligned}$$

In details:

- **Login:**

This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

$$1 \cdot w_{Simple,EI}$$

- **Logout:**

This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

$$1 \cdot w_{Simple,EI}$$

- **Registration:**

This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

$$1 \cdot w_{Simple,EI}$$

- **Handle Personal Profile:**

This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

$$1 \cdot w_{Simple,EI}$$

- **Taxi Reservation:**

This operation requires a medium effort. In fact it is very frequent.

$$1 \cdot w_{Medium,EI}$$

- **Taxi Request:**

This operation requires a medium effort. In fact it is very frequent.

$$1 \cdot w_{Medium,EI}$$

- **Notify Problem:**

This operation requires a strong effort. In fact it has to perform many elementary steps in order to handle the problem.

$$1 \cdot w_{Complex,EI}$$

- **End of Ride:**

This operation requires a medium effort. In fact it is very frequent.

$$1 \cdot w_{Medium,EI}$$

2.1.4 External Output

External Outputs are elementary operations that generate data for the external environment, and they usually include the elaboration of data from logic files. Summary of the calculated weights:

$$\begin{aligned} FP_{EO} &= 9 \cdot w_{Simple,EO} + 1 \cdot w_{Complex,EO} \\ &= 9 \cdot 4 + 1 \cdot 7 \\ &= 43 \end{aligned}$$

In details:

- **Logout:**

A redirection message needs to be sent to the user that logs out.

$$1 \cdot w_{Simple,EO}$$

- **Registration:**

The result of this operation must be sent to the user that registers

$$1 \cdot w_{Simple,EO}$$

- **Login:**

The result of the login must be sent to the specific user.

$$1 \cdot w_{Simple,EO}$$

- **Handle Personal Profile:**

Informations about the result of the modification must be sent to the related user

$$1 \cdot w_{Simple,EO}$$

- **Taxi Reservation:**

Notification about the result of the reservation must be sent to the specific passenger.

$$1 \cdot w_{Simple,EO}$$

- **Notify Problem:**

Notification about the result of the request must be sent to the specific passenger.

$$1 \cdot w_{Simple,EO}$$

- **Automatic Taxi Request:**

A new Taxi Request must be issued.

$$1 \cdot w_{Complex,EO}$$

- **Notification to the Passenger:**

Notification about the new Taxi Request must be sent to the specific passenger.

$$1 \cdot w_{Simple,EO}$$

- **Notification to the Taxi Driver:**

Notification about the result of the request must be sent to the specific Taxi Driver.

$$1 \cdot w_{Simple,EO}$$

- **End of Ride:**

Notification about the result of the operation must be sent to the relevant Taxi Driver.

$$1 \cdot w_{Simple,EO}$$

2.1.5 External Inquiry

External Inquiries are elementary operations that involve input and output, without significant elaboration of data from logic files.

Summary of the calculated weights:

$$\begin{aligned} FP_{EIQ} &= 2 \cdot w_{Simple,EIQ} \\ &= 2 \cdot 3 \\ &= 6 \end{aligned}$$

In details:

- **View Requests and Reservations:**

In order to perform these operations the system has only to retrieve, send and render simple data.

$$2 \cdot w_{Simple,EIQ}$$

2.1.6 Summary

All the calculated FP_i sums up to FP , which is the total Function Points value:

$$\begin{aligned} FP &= FP_{ILF} + FP_{ELF} + FP_{EI} + FP_{EO} + FP_{EIQ} \\ &= 73 + 17 + 30 + 43 + 6 \\ &= 169 \end{aligned}$$

The total FP value is then multiplied by a constant factor $k_{i,j}$ that depends on the programming language i used to develop the software and the company gearing ratio j .

The gearing ratio is the level of a company's debt related to its equity capital, usually expressed in percentage form. Gearing is a measure of a company's financial leverage and shows the extent to which its operations are funded by lenders versus shareholders.

This final calculation gives us the number of SLOC n_{SLOC} estimated for *myTaxiService*:

$$\begin{aligned} n_{SLOC} &= FP \cdot k_{Java,Avg} \\ &= 169 \cdot 53 \\ &= 8957 \text{ SLOC} \end{aligned}$$

2.2 COCOMO

This estimation method takes into account a lot of parameters, such as final software, personnel and platform characteristics, combining them using a complex non-linear process.

In order to generate the **Constructive Cost Model** we decided to use an online [COCOMO II calculator](#), using the **FP Sizing Method**.

We also used [COCOMO II - Model Definition Manual](#) to make better choices of the parameters we had to insert into the model.

- **Software Size**

- **Unadjusted Function Points:** The value FP has been taken as parameter, so **169** is the value chosen for this field.
- **Language:** The language of choice is **Java**, and not only Java EE, because the software is possibly a combination of different flavours of Java (Java SE, Java EE and Java for Android).

- **Software Scale Drivers**

- **Precedentedness:** Since we had a previous experience using *Java SE* for medium-size projects, but we had never used *Java EE* for developing such a big application we decided to set this parameter to **Nominal**.
- **Development Flexibility:** Given that we had not strict specifications we set this parameter to **High**.

- **Architecture/Risk Resolution:** Since we had designed several documents before the actual development, including this *PPD*, the development of the system to be has little chances of failing, so we choose a **Nominal** value.
- **Team Cohesion:** After a few days dedicated to create a efficient and fast workspace, the cohesion reached a **Very High** level.
- **Process Maturity:** We understand, support and follow the process so we choose a **High** level for this parameter.

- **Software Cost Drivers**

- **Product**


- * **Required Software Reliability:** Given that a failure in the software system could lead to moderate problems we choose **Nominal** level.
- * **Data Base Size:** Since we have a distributed application, the focus is on the lines of code instead of being on the size of the testing Database; so we choose a **Low** level parameter.
- * **Product Complexity:** We made an average of the various complexity areas and we choose a **High** level parameter.
- * **Developed for Reusability:** We decided to develop reusable system components, so we come up with an **High** level parameter.
- * **Documentation Match to Lifecycle Needs:** The standard level of documentation is required, so the chosen level is **Nominal**.

- **Personnel**

- * **Analyst Capability:** The personnel demonstrates a **Nominal** level of analysis ability.
- * **Programmer Capability:** The personnel demonstrates efficiency working together as a team, so we chose **High** level for this parameter.
- * **Personnel Continuity:** The project will be developed always by the initial programmers, so the *project's annual personnel turnover* is very low (7 % for year). For this reason we have chosen **High** for this parameter.
- * **Application Experience:** Since the last time the development team has worked on a so complicated project was a year ago, we have chosen **Nominal** for this parameter.
- * **Platform Experience:** The same as *Application Experience*; we have chosen **Nominal** level for this parameter too.
- * **Language and Toolset Experience:** The team is quite familiar with the development, analysis and design representation, so we choose **High** level for this parameter.

- **Platform**
 - * **Time Constraint:** We have no relevant time constraints, so we choose **Nominal** level for this parameter.
 - * **Storage Constraint:** We have no relevant storage constraint, so we choose **Nominal** level for this parameter.
 - * **Platform Volatility:** Our hardware and software platforms will not change often, so we will have no volatility and therefore we choose a **Low** level for this constraint.
- **Project**
 - * **Use of Software Tools:** The team is provided of a set of strong and mature life-cycle tools, moderately integrated one into each other. So we choose an **High** level for this parameter.
 - * **Multisite Development:** The team is in average fully collocated, so the chosen level is **Nominal**.
 - * **Required Development Schedule:** The project is not subjected on a particular constraint oppression, so we have chosen **Nominal** for this parameter.
- **Maintenance** This value is set to **Off**.
- **Software Labour Rates**
 - **Cost per Person-Month (Dollars):** We have chosen the average value of **1500\$/month** for this parameter.

Using the previous parameters, we have compiled the **COCOMO II Calculator** page:



COCOMO II - Constructive Cost Model

Software Size Sizing Method:

Unadjusted Function Points: Language:

Software Scale Drivers

Precedentedness: Architecture / Risk Resolution: Process Maturity:

Development Flexibility: Team Cohesion:

Software Cost Drivers

Product

Required Software Reliability: **Personnel**

Data Base Size: Analyst Capability:

Product Complexity: Programmer Capability:

Developed for Reusability: Personnel Continuity:

Documentation Match to Lifecycle Needs: Application Experience:

Platform Experience: Platform Volatility:

Language and Toolset Experience:

Platform

Time Constraint:

Storage Constraint:

Project

Use of Software Tools:

Multisite Development:

Required Development Schedule:

Maintenance

Software Labor Rates

Cost per Person-Month (Dollars):

Figure 1: COCOMO II Calculator - Parameters

And these are the results:

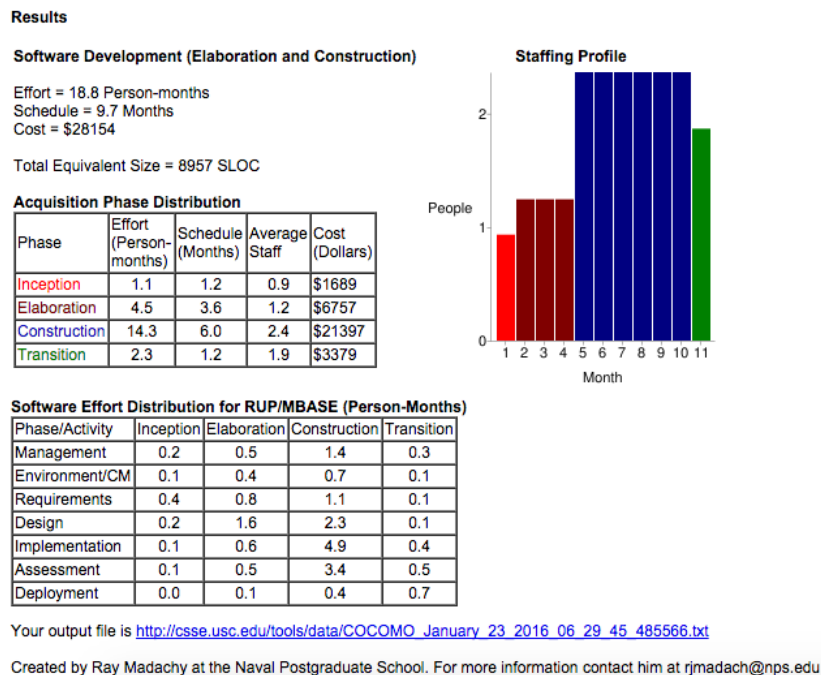


Figure 2: COCOMO II Calculator - Results

3 Project Tasks and Scheduling

3.1 Project Tasks

Several tasks has been identified in our project. The are summarized all together in the following table, which associates a label, a description and a completion state to each task:

Task	Description	Completed?
T1a	RASD - Writing	Yes
T1b	RASD - Presentation	Yes
T2a	DD - Writing	Yes
T2b	DD - Presentation	Yes
T3a	ITPD - Writing	Yes
T3b	ITPD - Presentation	Yes
T4a	PPD - Writing	Yes
T4b	Final Presentation	No
T5	Implementation	No
T6	Unit Testing	No
T7	Integration Testing	No
T8	System Testing	No
T9	User Acceptance - Alpha Testing	No
T10	User Acceptance - Beta Testing	No
T11	Release To Market	No

3.2 Tasks Scheduling

In this section is solved the scheduling problem for the identified tasks. We provide two tables and a diagram in order to have the most effective visualization of the scheduling.

3.2.1 Tasks, Durations and Dependencies

The following table summarizes for each task identified:

- The effort needed to perform such task (in terms of $[person \cdot hour]$),
- The work force (in terms of $[hour]$) each team component has available to do such task in the related period,
- The dependencies of the specific tasks to other tasks.

Task	Effort [<i>person · hour</i>]	Work Force [<i>hour</i>]	Dependencies
T1a	86	$22[day] \cdot 2[hour/day] =$ 44[hour]	
T1b	6	$1[day] \cdot 3[hour/day] =$ 3[hour]	T1a
T2a	74	$22[day] \cdot 2[hour/day] =$ 44[hour]	T1b
T2b	6	$1[day] \cdot 3[hour/day] =$ 3[hour]	T2a
T3a	30	$14[day] \cdot 1.5[hour/day] =$ 21[hour]	T2b
T3b	6	$1[day] \cdot 3[hour/day] =$ 3[hour]	T3a
T4a	32	$10[day] \cdot 2[hour/day] =$ 20[hour]	T3b
T4b	10	$1[day] \cdot 3[hour/day] =$ 5[hour]	T4a
T5	480	$80[day] \cdot 3[hour/day] =$ 240[hour]	T4b
T6	160	$20[day] \cdot 4[hour/day] =$ 80[hour]	T5
T7	140	$20[day] \cdot 4[hour/day] =$ 80[hour]	T6
T8	70	$20[day] \cdot 2[hour/day] =$ 40[hour]	T7
T9	80	$20[day] \cdot 2[hour/day] =$ 40[hour]	T8
T10	80	$20[day] \cdot 2[hour/day] =$ 40[hour]	T9
T11	8	$1[day] \cdot 4[hour/day] =$ 4[hour]	T10

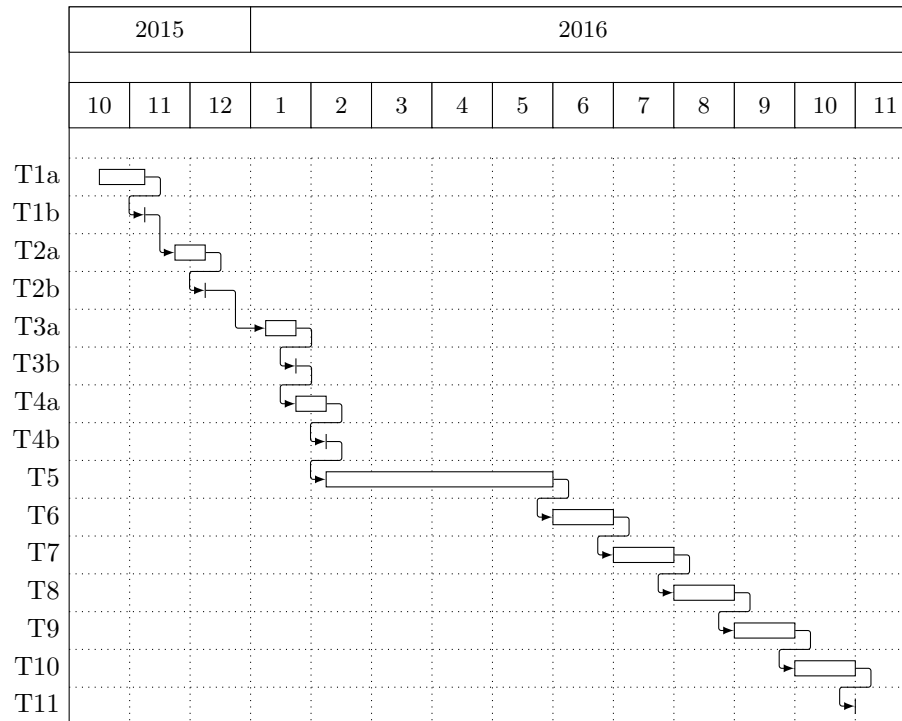
Then, for each task, are identified:

- The date in which the given task starts,
- The date in which the given task ends,
- The interval in [*day*] that separates the starting date from the ending date.

Task	Start	End	Interval
T1a	15/10/2015	6/11/2015	22
T1b	11/11/2015	11/11/2015	1
T2a	12/11/2015	4/12/2015	22
T2b	9/12/2015	9/12/2015	1
T3a	07/01/2016	21/01/2016	14
T3b	22/01/2016	22/01/2016	1
T4a	22/01/2016	02/02/2016	10
T4b	03/02/2016	03/02/2016	1
T5	04/02/2016	30/05/2016	117
T6	31/05/2016	29/06/2016	29
T7	30/06/2016	28/07/2016	28
T8	29/07/2016	29/08/2016	31
T9	30/08/2016	27/09/2016	28
T10	28/09/2016	26/10/2016	28
T11	27/10/2016	27/10/2016	1

3.2.2 Gantt Diagram

Here is built a Gantt Diagram showing the schedule chosen for *myTaxiService* project tasks.



4 Resources Allocation

This section covers the problem of allocating the human resources to each task in order to respect the identified scheduling. We are a group of two people, and we do not have any kind of concurrency in the tasks that we identified, so we are going to work together on the same task at the same time during the whole duration of the project. As shown in the [result of the COCOMO II - Calculator](#) in the correspondent section, the project can be managed by more or less two people in nine months, and this respects the scheduling we devised.

4.1 Staff Allocation Chart

A chart is provided for the best visualization of the staff allocation. For graphical issues, some consecutive tasks are merged one into the other: T1a and T1b are merged into T1, T2a and T2b are merged into T2, T3a and T3b are merged into T3, T4a and T4b are merged into T4, T10 and T11 are merged into T12.

		2015			2016										
		10	11	12	1	2	3	4	5	6	7	8	9	10	11
Alberto		T1	T2		T3	T4	T5			T6	T7	T8	T9	T12	
Alessandro		T1	T2		T3	T4	T5			T6	T7	T8	T9	T12	

5 Project Risks

In order to adopt a global **Proactive Risk Management Strategy**, we identified all the possible risks that could affect *myTaxiService*. Then, for each risk, we analysed the probability that it will occur and the impact that it will have on the project if it does occur. Furthermore, a contingency plan has been devised for all the identified risks.

5.1 Risk Types

All the risk types identified are here reported:

- **Project Risks**

1. Members of the team are ill at critical times in the project.
 - **Probability:** Moderate
 - **Effect:** Serious
 - **Management:** Organize the team so that there is more overlap of work and each team member for this reason understands each other's job.
2. The work force available is less than expected.
 - **Probability:** Moderate
 - **Effect:** Serious
 - **Management:** In scheduling phase, take into account that possibility and assign a bit more effort than needed to each task.

- **Technical Risks**

1. The database and other external software components used in the system cannot process as many operations per second as expected.
 - **Probability:** Moderate
 - **Effect:** Serious
 - **Management:** Overestimate the operations throughput in order to choose the appropriate support infrastructure.
2. Faults in reusable software components have to be repaired before these components are reused.
 - **Probability:** Moderate
 - **Effect:** Serious
 - **Management:** Develop strong unit tests so that the error probability is reduced and in the worst case the reparation time is minimized.

- **Business Risks**

- **Market Risk:**

1. The project, once developed, demonstrates a complete misunderstanding of the requirements.
 - * **Probability:** Low
 - * **Effect:** Catastrophic
 - * **Management:** Design a complete RASD and submit it to peer reviewing.

- **Strategic Risk:**

1. The business strategy changes and *myTaxiService* is no longer a priority for the company.
 - * **Probability:** Low
 - * **Effect:** Catastrophic
 - * **Management:** Prepare a briefing document for the business management showing how the project is going to provide concrete results, given the high market value of *myTaxiService*.

- **Sales Risk:**

1. The project, once developed, comes out as too complex and not easily marketable.
 - * **Probability:** Moderate
 - * **Effect:** Catastrophic
 - * **Management:** Design a DD with a high level of detail, and double check it against the RASD.
2. The user interfaces developed are not user-friendly.
 - * **Probability:** Low
 - * **Effect:** Moderate
 - * **Management:** Follow the guidelines provided by the designers for developing modern, thin and usable UIs.

- **Management Risk:**

1. The company changes the management group and the new one decides to change the project focus or the project work force.
 - * **Probability:** High
 - * **Effect:** Serious
 - * **Management:** Prepare a briefing document for management group showing how *myTaxiService* is making important contributions to the business goals.

- **Budget Risk:**

1. The company decides to reduce the budget for the development of *myTaxiService*.
 - * **Probability:** Low

- * **Effect:** Catastrophic
 - * **Management:** Prepare a briefing document for management showing how cutting the budget for the project would not be cost-effective.
2. A developer of *myTaxiService* decide to stop the development for an overlapping with other personal commitments.
- * **Probability:** Low
 - * **Effect:** Serious
 - * **Management:** Investigate the possibility of buying-in some software components, and the possibility of hiring new team members.

6 Appendix

6.1 Tools

- **TeXstudio:** L^AT_EX editor used to write this document.
- **SourceTree:** Used to allow team work and synchronize GitHub repository.

6.2 Hours of Work

- **Alessandro:** 16
- **Alberto Mario:** 16

6.3 Revision History

<i>Version Number</i>	<i>Release Date</i>	<i>Changelog</i>
1.0	24/01/2016	Initial Release.