

Politecnico di Milano
Scuola di Ingegneria dell'Informazione
Corso di Laurea Magistrale in Computer Science and Engineering
A.Y. 2015-2016



Software Engineering 2 Project
“myTaxiService”
Project Plan

February 1, 2016

Principal Adviser: Prof. **Di Nitto**

Authors:

Davide Fisicaro 854043

Gianmarco Giummarra 852667

Salvatore Ferrigno 850130

CONTENTS

Contents

1	Introduction	1
2	Function Points	2
2.1	Function Points Estimation	3
2.1.1	Internal Logic Files	3
2.1.2	External Logic Files	3
2.1.3	External Inputs	4
2.1.4	External Outputs	5
2.1.5	External Inquiries	5
2.2	Function Points Summary	6
3	COCOMO II Approach	7
4	Tasks and Schedule	10
4.1	Identified Tasks	10
4.1.1	RASD	10
4.1.2	Design Document	10
4.1.3	Implementation	10
4.1.4	Integration and System Testing	11
4.2	Project Schedule	11
5	Risks	12
5.1	Project risks	12
5.2	Technical risks	12
5.3	Business risks	12
6	Used Tools	13

1 Introduction

The aim of this document is to provide an estimation of the cost that the project will take in terms of time, required lines of code, effort, duration and number of people needed.

The way to do this consists in analyzing all the required functionalities of the system to be developed and evaluating, one by one, the complexity of that operations basing on the number of required data inputs and outputs. After doing this a first evaluation of the required number of code lines can be made using specific tools, while the estimated time and effort required will be provided using the COCOMO solution.

After the cost estimation can be found an identification of the tasks to accomplish in order to develop the system, and the project plan that identifies the tasks execution order. The definition of the tasks allows to do an initial allocation of the human resources composing the working team.

Finally a risk analysis is made in order to identify the principal problems that can be detected during the project development: such analysis made a priori allows the formulation of a contingency plan that will lead to a risk avoidance and will help to solve problems in the eventuality of their appearance.

2 Function Points

Function points can be divided into five categories, basing on the type of considered data:

- **ILF - Internal Logic File:** homogeneous set of data used and managed by the application;
- **ELF - External Logic File:** homogeneous set of data used by the application but generated and maintained by other applications;
- **External Input:** elementary operation to elaborate data coming from the external environment;
- **External Output:** elementary operation that generates data for the external environment. Also includes the elaboration of data from logic files;
- **External Inquiry:** elementary operation that involves input and output. No significant elaboration of data from logic files is needed.

For every type of function point the estimation will take into account the complexity of the operation, using the following weight table:

<i>Function Type</i>	<i>Complexity</i>		
	<i>Simple</i>	<i>Medium</i>	<i>Complex</i>
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

2.1 Function Points Estimation

2.1.1 Internal Logic Files

The application includes several internal logic files that are used to store information about Customer, Taxi Driver, Generic Call, Ride and Notification. Each of these entities has a simple structure as it is composed by a small number of fields except for the Generic Call that has a more complex structure and needs more operations during the computation (path calculation, search for an available taxi driver, ride matching for the shared ones, notification to the selected taxi driver and so on). Therefore it is adopted a complex weight for Generic Call and a simple weight for all other entities. The total amount of function points concerning internal logic files is: $(1 \times 15) + (4 \times 7) = 43$

2.1.2 External Logic Files

The application has to use ELFs for execute several operations:

- acquisition of the city map and the positions of both Taxi driver and customer. These data are obtained using the chosen GPS APIs;
- acquisition of users information through the Google+ service;
- acquisition of users information through the Facebook service;
- interaction with an external banking service. Such service sends to the system all the information about the validity of the customers payment information and the data related to a specific payment.

Due to the complexity of these interactions (acquire city map, retrieve a position, interface with the banking system) and the structure of the received data , it's reasonable to choose a medium weight. $3 \times 7 + 2 \times 5 = 31$

2.1.3 External Inputs

- *Login/Logout*: these are simple operations, so it is adopted the simple weight for them. $2 \times 3 = 6$

1. Customer

- *Registration*: the guest fills the registration form, this is a simple operation, so it is adopted the simple weight for it. $1 \times 3 = 3$
- *Rate a Taxi Driver*: this is a simple operation, so we can adopt the simple weight for it. $1 \times 3 = 3$
- *Make an Immediate Call*: this is not a simple operation. It involves several components: ImmediateCallSystem, TaxiDriver, DataManager, PaymentManager. For this reason it is adopted the medium weight for it. $1 \times 4 = 4$
- *Make a Reservation Call*: it is an operation that involves the same components of the immediate call operation, so it is adopted the medium weight for it as before. $1 \times 4 = 4$
- *Make a Shared Call*: this operation involves the same components involved in the other "Call" operation but it is more complex, so it is adopted the complex weight for it. $1 \times 6 = 6$
- *Update personal informations*: it is a simple operation, so it is adopted the simple weight. $1 \times 3 = 3$
- *Insert/Update payment method*: these are simple operations, it is adopted the simple weight for them. $2 \times 3 = 6$

2. Taxi Driver

- *Assistance Call*: this is a simple operation because involves only the "AssistanceCallManager" component, for this reason it is adopted the simple weight for it. $1 \times 3 = 3$
- *Update personal informations*: it is a simple operation, so it is adopted the simple weight for it. $1 \times 3 = 3$
- *Accept/Decline a call request*: these are simple operation, so it is adopted the simple weight for them. $2 \times 3 = 6$

3. Admin

- *Add new taxi driver*: this is not a simple operation. It is adopted a medium weight for it. $1 \times 4 = 4$
- *Manage User Data*: this is a not simple operation, it is adopted a medium weight for it. $1 \times 4 = 4$

2.1.4 External Outputs

- *Request of payment to the banking system and Notify the success or the failure of the operation to the Customer:* for this operation it is adopted a medium weight. $1 \times 5 = 5$
- *Notify the confirmation or the rejection of a Call request:* this is a simple operation, it is adopted a simple weight. $1 \times 4 = 4$
- *Compute the path of a specific ride and communicate it to the Taxi Driver:* this is a complex operation for this reason is adopted a complex weight. $1 \times 7 = 7$
- *Searching for an available taxi Driver and sending a notification to him:* this is not a simple operation, it is adopted a medium weight. $1 \times 5 = 5$
- *After registration the success or the failure of the operation is notified to the user:* this is a simple operation, so it is adopted a simple weight. $1 \times 4 = 4$
- *After an Assistance Call, the system sends a notification to the correspondent technical service:* this is a simple operation, so it is adopted a simple weight. $1 \times 4 = 4$

2.1.5 External Inquiries

- *Request to consult rides history:* the system allows both taxi driver and Customer to consult the Ride History, it is a simple operation, so it is adopted a simple weight. $1 \times 3 = 3$
- *Consult personal information:* the system allows every type of user to consult his personal information. This is a simple operation, for this reason it is adopted a simple weight. $1 \times 3 = 3$
- *Request to consult payments history:* it is a simple operation, so it is adopted a simple weight. $1 \times 3 = 3$

2.2 Function Points Summary

The analysis made in the previous section can be summed up in the following table:

<i>Function Type</i>	<i>Value</i>
Internal Logic File	43
External Logic File	31
External Input	55
External Output	29
External Inquiry	9
<i>Total</i>	167

The total value can be used as the basis to estimate, using the Average Variable Cost (AVC) corresponding to the Java language in the tables¹, which corresponds to 53.

The resulting number of lines in this case is:

$$167 \text{ FPs} * 53 = 8851 \text{ SLOC} = 8.851 \text{ KSLOC (1)}$$

¹<http://www.qsm.com/resources/function-point-languages-table/>

3 COCOMO II Approach

The COCOMO II tool has been used in order to compute effort and duration of the project development, starting from the number of identified FPs.

In the first step of the estimation all the Cost and Scale Drivers have been considered as “Nominal”, so the resulting EAF and exponent value E are:

- Effort Adjustment Factor, derived from Cost Drivers:

$$EAF = \prod_i CD_i = 1.00;$$

- Exponent, derived from Scale Drivers:

$$E = 0.91 + 0.01 * \sum_i SF_i = 1.0997;$$

At this point, following the related formula of the model

$$effort = 2.94 * EAF * (KSLOC)^E \quad (2)$$

and substituting the values obtained with the formula (1) and the parameters described above, the resulting *effort* value is

$$effort = 2.94 * (1.0) * (8.851)^{1.0997} = 32.341 \text{ Person-Months}$$

Through the found effort value, it's possible to calculate the value of the duration (in months) of the project, using an exponent

$$E' = 0.28 + 0.2 * (E - 0.91) = 0.3179$$

with the formula

$$Duration = 3.67 * (effort)^{E'} \quad (3)$$

obtaining

$$Duration = 3.67 * 32.341^{0.3179} = 11.082 \text{ Months}$$

At this point the number of people needed, N, is easy to find

$$N = \frac{effort}{Duration} \quad (4)$$


obtaining, substituting the values,

$$N = \frac{32.341 \text{ PM}}{11.082 \text{ Months}} = 2.92 \simeq 3 \text{ Persons}$$

which, in first approximation, it's consistent with the number of people involved in this project (see *Section 4* of this document for further information).

At this point of the evaluation it's good practice to adjust and customize some Cost and Scale Drivers, in order to give a more precise evaluation of the parameters; in order to do so is used the COCOMO II tool², to comfortably select the value for each driver and automatically change the equations parameters. The results of the execution of the tool are shown in the screenshots above.

²<http://csse.usc.edu/tools/COCOMOII.php>



COCOMO II - Constructive Cost Model

Software Size Sizing Method

Unadjusted Function Points Language

Software Scale Drivers

Precedentedness	<input type="text" value="Nominal"/>	Architecture / Risk Resolution	<input type="text" value="High"/>	Process Maturity	<input type="text" value="Nominal"/>
Development Flexibility	<input type="text" value="Nominal"/>	Team Cohesion	<input type="text" value="Extra High"/>		

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	<input type="text" value="Low"/>	Analyst Capability	<input type="text" value="Nominal"/>	Time Constraint	<input type="text" value="Very High"/>
Data Base Size	<input type="text" value="Nominal"/>	Programmer Capability	<input type="text" value="High"/>	Storage Constraint	<input type="text" value="High"/>
Product Complexity	<input type="text" value="Nominal"/>	Personnel Continuity	<input type="text" value="Very High"/>	Platform Volatility	<input type="text" value="Low"/>
Developed for Reusability	<input type="text" value="Nominal"/>	Application Experience	<input type="text" value="Low"/>	Project	
Documentation Match to Lifecycle Needs	<input type="text" value="Nominal"/>	Platform Experience	<input type="text" value="Low"/>	Use of Software Tools	<input type="text" value="Nominal"/>
		Language and Toolset Experience	<input type="text" value="Low"/>	Multisite Development	<input type="text" value="Nominal"/>
				Required Development Schedule	<input type="text" value="Nominal"/>

Maintenance

Software Labor Rates

Cost per Person-Month (Dollars)

Figure 3.1: COCOMO 2 drivers screen

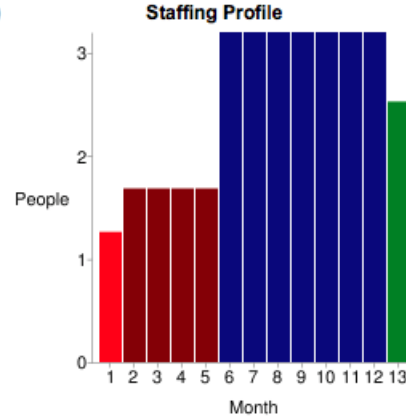
Results**Software Development (Elaboration and Construction)**

Effort = 29.5 Person-months
 Schedule = 11.2 Months
 Cost = \$58962

Total Equivalent Size = 8851 SLOC

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.8	1.4	1.3	\$3538
Elaboration	7.1	4.2	1.7	\$14151
Construction	22.4	7.0	3.2	\$44811
Transition	3.5	1.4	2.5	\$7075

**Software Effort Distribution for RUP/MBASE (Person-Months)**

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.2	0.8	2.2	0.5
Environment/CM	0.2	0.6	1.1	0.2
Requirements	0.7	1.3	1.8	0.1
Design	0.3	2.5	3.6	0.1
Implementation	0.1	0.9	7.6	0.7
Assessment	0.1	0.7	5.4	0.8
Deployment	0.1	0.2	0.7	1.1

Figure 3.2: COCOMO 2 results screen

According to the COCOMO II results

$$effort = 29.5 \text{ PM}$$

$$Duration = 11.2 \text{ Months}$$

$$N = \frac{effort}{Duration} = \frac{29.5 \text{ PM}}{11.2 \text{ Months}} = 2.634 \simeq 3 \text{ Persons}$$

The results are similar as in the case of Nominal Drivers, due to the balancement of the values of certain drivers.

4 Tasks and Schedule

This section defines the tasks to be accomplished in order to fulfill the goal of the project and the schedule that will define the work that every team component will have to perform. The chosen model for the project is the *Waterfall Model* which follows a linear execution of the various tasks from the first to the last task.

The team is composed of three Junior developers, called for simplicity Member A, Member B and Member C; due to the limited number of components the decisional issues can be managed by the whole team.

The tasks listed below are of two kinds: executable by a single member and executable by a group of collaborating members. Some of the tasks need to be executed in parallel with other ones, in this case the greatest number of members has to be assigned to the most critical task.

The list doesn't reflect strictly the execution sequence of the tasks.

4.1 Identified Tasks

4.1.1 RASD

- Requirements collection
- Requirements analysis
- Requirements formalization
- Models formalization
- Alloy Modeling

4.1.2 Design Document

- Architectural Design
- UI Design
- Algorithm Design
- Requirements traceability
- Requirements data management

4.1.3 Implementation

- Code implementation
- Unit Testing
- Code inspection

4.1.4 Integration and System Testing

- Integration strategy
- Integration test planning
- System testing

An intermediate review is to be done after each one of the four steps, and a final review must be done before the release of the application.

4.2 Project Schedule

In order to clearly describe the project schedule it's shown below a Gantt Diagram, in which are explicated also the Resource Allocations, related to Members A, B and C.

The starting date of the project is October 1st, 2015, while the estimated end date is September 30th, 2016.

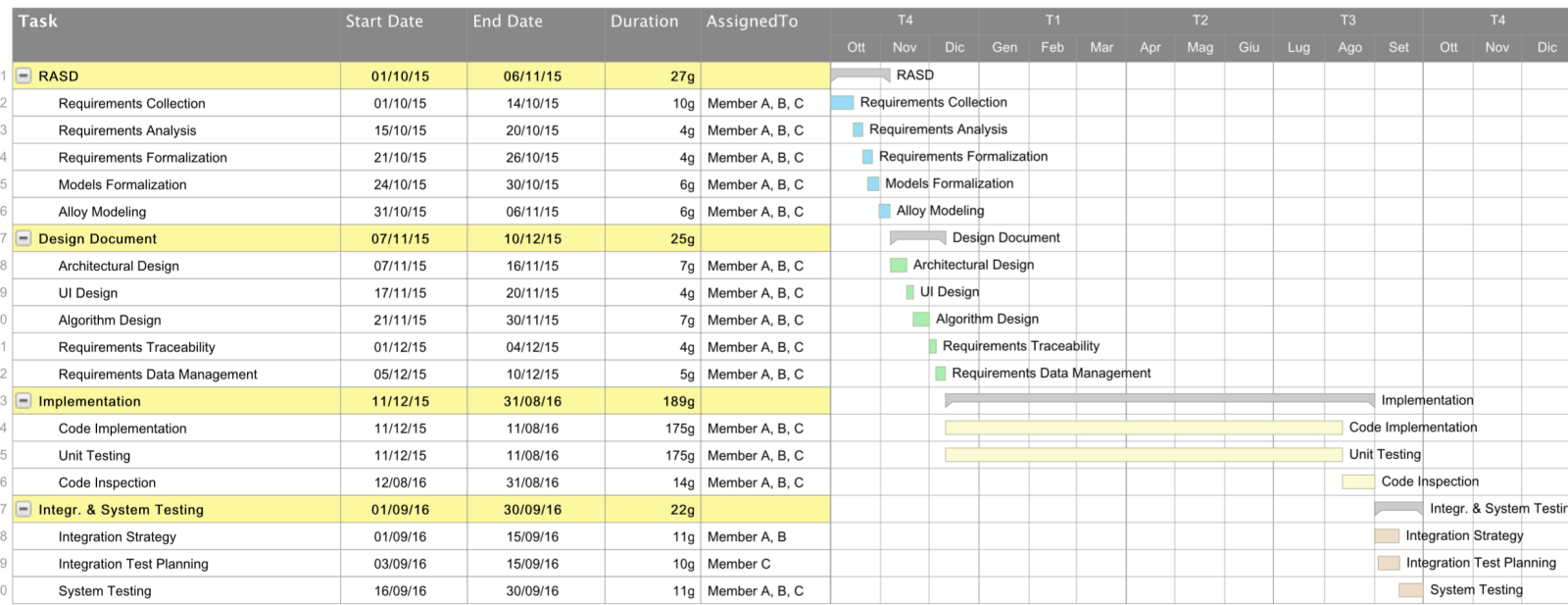


Figure 4.1: Gantt Diagram for the project schedule

5 Risks

Analysing the risks that could occur during the project development, three categories have been defined.

5.1 Project risks

One of the main risks that could threaten this project is the members management, due to the fact that there is not a hierarchy and every decision is made after a brainstorming in which all the members are involved.

Another issue, due to the limited team dimension, is that the project plan deadlines could not be respected for several issues: illness or any kind of unexpected unavailabilities. The way to avoid this is trying to maximize the collaboration of the task execution, in order to have a sure coverage of each task.

A third problem could be the withdrawal of one or more members, which could be resolved with the assumption of a new member or an extension of the deadlines. Although it is considered by the COCOMO II tool, it's a risk to be taken into account because of the facts that the members continuity is not a predictable driver. A way to minimize the damage caused if this risk becomes real is to ask the members to notice in time the eventual withdrawal and to have some contact of eventually available substitutes.

Another potential risk is that, at a certain point of the development course, the stakeholders want to add or change some functionalities or requirements; this could significantly affect the duration and the effort of the first part of the project, depending on the entity of the changes. It's possible to try to avoid this risk giving particular importance to the first phase of the project, discussing exhaustively all the requirements before starting to model the application.

5.2 Technical risks

A potential technical risk to be taken into account is the loss of data which the team is working on. The most efficient way to avoid or face up this risk is to use Distributed Version Control tools like *git* and, eventually, the support of Cloud tools. Backups are to be done periodically, i.e. at the end of each working day.

5.3 Business risks

The main business risk is that the Government of the large city that committed the work changes its mind about the project to be built and stops funding the team. In order to avoid this unwanted situation, an analysis of the stakeholder's reliability could be done by the team.

6 Used Tools

- *LyX* to generate the pdf document
- *COCOMO II*³ to estimate effort, duration and number of members needed
- *Smartsheet*⁴ to generate the Gantt Diagram

³<http://csse.usc.edu/tools/COCOMOII.php>

⁴<http://www.smartsheet.com>