POLITECNICO
MILANO 1863

A.Y 2015-2016
Software Engineering 2: "myTaxiService"

Project **P**lan

Version 1.0

Massimiliano Paci (mat. 852720)
Giovanni Patruno (mat. 852658)

February 2nd 2016

Contents

1 Introduction **4**
- 1.1 Revision History . . . 4
- 1.2 Purpose . . . 4
- 1.3 Scope . . . 4
- 1.4 Definitions, Acronyms, Abbreviations . . . 4
  - 1.4.1 Acronyms . . . 4
- 1.5 List of Reference Documents . . . 5

2 Size, Cost and Effort Estimation **6**
- 2.1 Overview . . . 6
- 2.2 Size Estimation: Function-Points . . . 6
  - 2.2.1 ILF - Internal Logical Files . . . 6
  - 2.2.2 EIFs - External Interface Files . . . 6
  - 2.2.3 EIs - External Inputs . . . 7
  - 2.2.4 EIQs - External Inquires . . . 7
  - 2.2.5 EOs - External Outputs . . . 7
  - 2.2.6 UFP - Un-adjusted Function-Points . . . 7
- 2.3 COCOMO - COnstructive COst MOdel . . . 8
  - 2.3.1 Software Scale Drivers . . . 8
  - 2.3.2 Software Cost Drivers - Product . . . 8
  - 2.3.3 Software Cost Drivers - Personnel . . . 8
  - 2.3.4 Software Cost Drivers - Platform . . . 9
  - 2.3.5 Software Cost Drivers - Project . . . 9

3 Tasks **12**
- 3.1 Design phase's tasks identified . . . 12
- 3.2 Development phase's tasks identified . . . 13
- 3.3 Task Scheduling . . . 13
  - 3.3.1 RASD Scheduling . . . 13
  - 3.3.2 DD Scheduling . . . 14
  - 3.3.3 Integran Test Plan Scheduling . . . 14
  - 3.3.4 Project Planning Scheduling . . . 14
  - 3.3.5 Development Scheduling . . . 15

4 Project's risks **16**
- 4.1 Project Risks . . . 16
  - 4.1.1 Team's availability . . . 16
  - 4.1.2 Error in the requirements . . . 16
- 4.2 Technical Risks . . . 16
  - 4.2.1 Server dimension . . . 16
  - 4.2.2 Platform damaged . . . 16
  - 4.2.3 Application's bug . . . 17
- 4.3 Business Risks . . . 17
  - 4.3.1 Possible competitors . . . 17

5 Appendix **18**
5.1 Software and tool used 18
5.2 Working Hours 18

1 Introduction

1.1 Revision History

02/02/2016 : Version 1.0

1.2 Purpose

The purpose of the Project Plan Document is to estimate effort, cost, size and risks of myTaxiService application and with these try to elaborate a schedule of the development work

1.3 Scope

The aim of this project is to create a platform, named **myTaxiService**, to optimize the taxi service of a large city.

Passengers will be able to book a taxi for a certain route, view a cost preview and the relative waiting time.

Taxi drivers will be able to inform the system about their availability in a certain zone, confirm or deny a ride and consult the feedback of the passengers.

The city is divided in taxi zones and each one is associated to a queue of taxis. The system assigns every booking to an available driver in the ride's zone. If there is not a driver available in that specific zone, the system automatically assigns another available driver from another zone.

Taxi drivers will have access to the platform too: they will be able to check their queue state and and also see statistics on their routes.

The platform will be scalable and it could be integrated with other applications to implement new functionalities.

1.4 Definitions, Acronyms, Abbreviations

1.4.1 Acronyms

- DD: Design Document
- RASD: Requirement Analysis and Specification Document
- ILF: Internal Logical Files
- EIF: External Interface Files
- EI: External Inputs
- EIQ: External Inquires
- EO: External outputs
- UFP: Un-adjusted Function-Points
- COCOMO: COnstructive COst MOdel
- ITPD: Integration Test Plan Document

1.5 List of Reference Documents

- Project's assignments
- Principles of FP Analysis - http://www.softwaremetrics.com/Function%20Point%20Training%20Booklet%20New.pdf
- COCOMO II Model Definition Manual http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

2 Size, Cost and Effort Estimation

2.1 Overview

In this section we will show an estimation of the size of project using the algorithmic approach of the Fuction-Points(FP); after these we will calculate an estimation of cost and effort with COCOMO II.

2.2 Size Estimation: Function-Points

We will present the function-points divided by types specifying a prevision of complexity for each one.
In the following tables there are the weights for each function type:

Function Types		**Weight**	
	Simple	Medium	Complex
N. Inputs	3	4	6
N. Outputs	4	5	7
N. Inquiry	3	4	6
N. ILF	7	10	15
N. EIF	5	7	10

2.2.1 ILF - Internal Logical Files

The system stores the information about:

- Users - *Simple*
- Bookings - *Medium*
- Rides - *Simple*
- Feedbacks - *Simple*

All these entities, except *Bookings*, have a simple structure of limited number of fields, so we can use the simple weight.

3x7 + 1x10 = 31 FPs

2.2.2 EIFs - Esternal Interface Files

The application manages informations given from entity external from the system:

- OpenStreetMap API provides information of zones and traffic - 2x *Medium*
- Taxi Informations: GPS location, Taxi Availability - 2x *Simple*

The entity provides by the map API are of medium complexity but the taxi informations are very simple.

2x5 + 2x7 = 24 FPs

2.2.3 EIs - External Inputs

The application interacts with the user as follow:

- Login/Logout - 2x *Simple*
- Create/modify/delete booking - 3¡*Complex*
- Feedback Releasing - *Complex*
- Release taxi's availability - *Simple*
- Confirm/refuse Ride - *Complex*

Login, Logout and release taxi's availability are simple operations. The others involve more entities (users, bookings, zones, ...) so are considered complex.

3x3 + 3x6 = 27 FPs

2.2.4 EIQs - External Inquires

The application allow users to:

- Visualize profiles - *Simple*
- Visualize bookings - *Medium*
- Visualize feedbacks - *Medium*

1x3 + 2x4 = 11 FPs

2.2.5 EOs - External Outputs

The application generates the following data for the external environment

- User's notifications - *Medium*
- API provide for external applications -*Complex*

1x5 + 1x7 = 12 FPs

2.2.6 UFP - Un-adjusted Function-Points

In conclusion adding all the FPs' values we obtained a first measure of the complexity of the application:
31 + 24 + 27 + 11 + 12 = 105 UFPs

Considering that we will use COCOMO for further estimations, we accept the UFP without adjustments.

2.3 COCOMO - COnstructive COst MOdel

To calculate Effort, Duration and Cost first we have assign a value to each element of the COCOMO model that corrispond to a particular multiplicator. Then we have used an online tool for the calculus (`http://csse.usc.edu/tools/COCOMOII.php`).

2.3.1 Software Scale Drivers

- Precedentedness - *Very Low* - The team doesn't have previous experience in this kind of project
- Development Flexibility - *High* - The client doesn't set very rigorous rules for the development process
- Achitecture / Risk Resolution - *Nominal* - An average risk analysis
- Team Cohesion - *Very High* - The team members know very well each other and they work together several times
- Process Maturity - *Nominal*

2.3.2 Software Cost Drivers - Product

- Required Software Reliability - *High* - We need an high reliability for the system
- Database Size - *Nominal*
- Product Complexity - *High* - The overall complexity of the project is quite high
- Developed for Reusability - *High*
- Documentation Match to Lifecycle Needs - *High*

2.3.3 Software Cost Drivers - Personnel

- Analyst Capability - *Nominal*
- Programmer Capability - *High*
- Personnel Continuity - *Low*
- Application Experience - *Low*
- Platform Experience - *Low*
- Language and Toolset Experience - *High*

2.3.4 Software Cost Drivers - Platform

- Time Constraint - *Nominal* - We want that the system respond in an average time
- Storage Constraint - *Nominal*
- Platform Volatility - *Nominal*

2.3.5 Software Cost Drivers - Project

- Use of Softare Tools -*High*
- Multisite Development - *Very Low*
- REquired Development Schedule - *High*

Software Size Sizing Method: Function Points

Unadjusted Function Points: 105 Language: Java

Software Scale Drivers

Precedentedness	Very Low	Architecture / Risk Resolution	Nominal	Process Maturity	Nominal
Development Flexibility	High	Team Cohesion	Very High		

Software Cost Drivers

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

Maintenance Off

Software Labor Rates

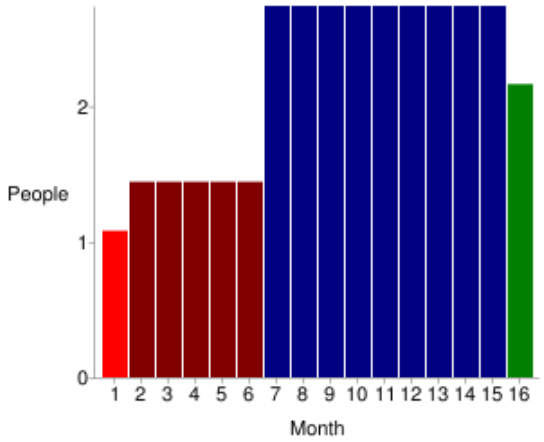
Cost per Person-Month (Dollars): 1500

Software Development (Elaboration and Construction)

Effort = 34.6 Person-months
Schedule = 15.4 Months
Cost = $51916

Total Equivalent Size = 5565 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	2.1	1.9	1.1	$3115
Elaboration	8.3	5.8	1.4	$12460
Construction	26.3	9.6	2.7	$39457
Transition	4.2	1.9	2.2	$6230

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

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.3	1.0	2.6	0.6
Environment/CM	0.2	0.7	1.3	0.2
Requirements	0.8	1.5	2.1	0.2
Design	0.4	3.0	4.2	0.2
Implementation	0.2	1.1	8.9	0.8
Assessment	0.2	0.8	6.3	1.0
Deployment	0.1	0.2	0.8	1.2

3 Tasks

In the lists below you will find the tasks that will be presented in a detailed Gantt diagram.

3.1 Design phase's tasks identified

- Team creation - choosing of the team's members
- Overview of the product constraint - concerning with the RASD
- Overview of the product constraint
- Overview of the specific requirements
- RASD Introduction - writing
- RASD Overall description - writing
- RASD Specific requirements - writing
- RASD Alloy - creation of the alloy model
- Selection of the architecture design
- Design document Introduction
- Design document architectural design
- Design Document algorithm design
- Design Document User interface design
- Design Document Requirements traceability
- Design Document Requirements traceability
- Selection of the strategy for the integration test plan
- ITPD Introduction
- ITPD Integration Strategy
- ITPD Individual steps and test description
- ITPD Tools and test equipment required
- Function points calculus - points used in the COCOMO model constructed
- COCOMO II model
- Tasks description
- Results analysis

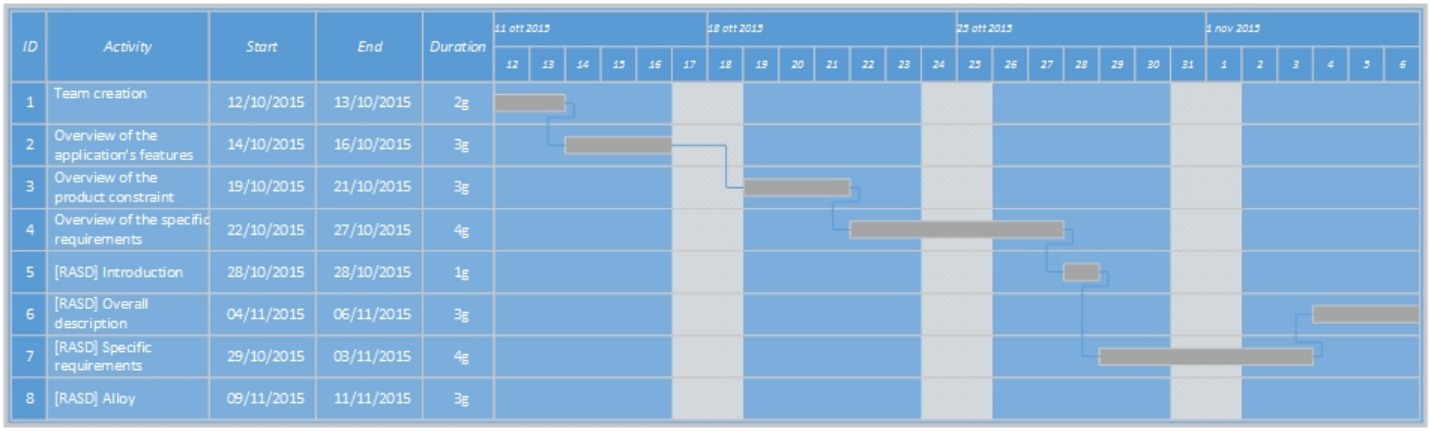
3.2 Development phase's tasks identified

- Development environment setup - setting up the environment on which programmers and analyst will work
- API and external tools setup
- Develop the different Host-side components with a bottom-up approach
- Develop the different Guest-side components with a bottom-up approach
- Unit testing phase of the project
- Final debug
- Bug fix
- Servlet deployment - Deployment of Servlet on the host-machine

3.3 Task Scheduling

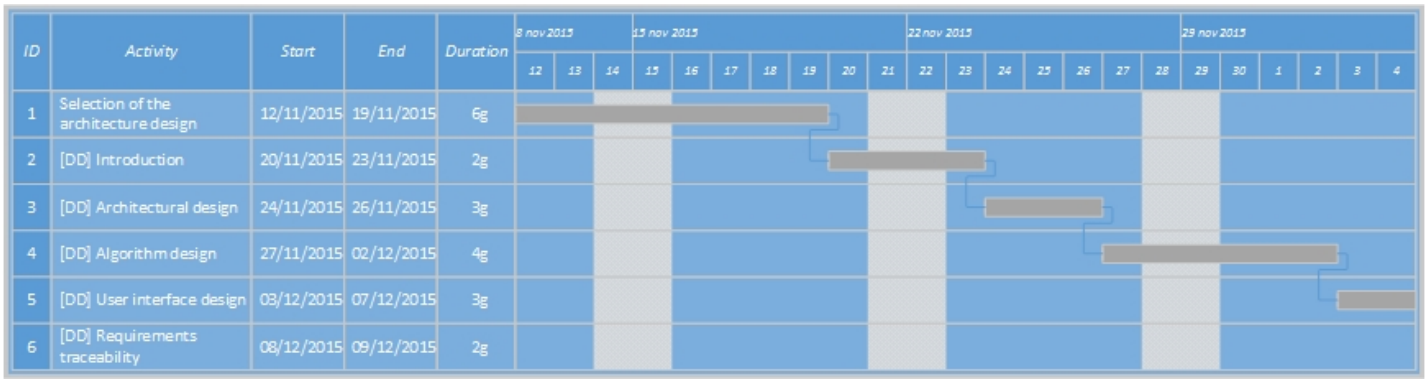
For this scheduling we skipped the task assignment because of the team composition, we are two people and we work every time together because of a logistic reason, so this task will be developed in parallel between the two members: Paci Massimiliano, Patruno Giovanni.
The task scheduling is split in four sections that are the different purposes of the work (RASD, DD, ITPD, Planning and Development) in order to keep the Gantt diagram clear and suitable.

3.3.1 RASD Scheduling



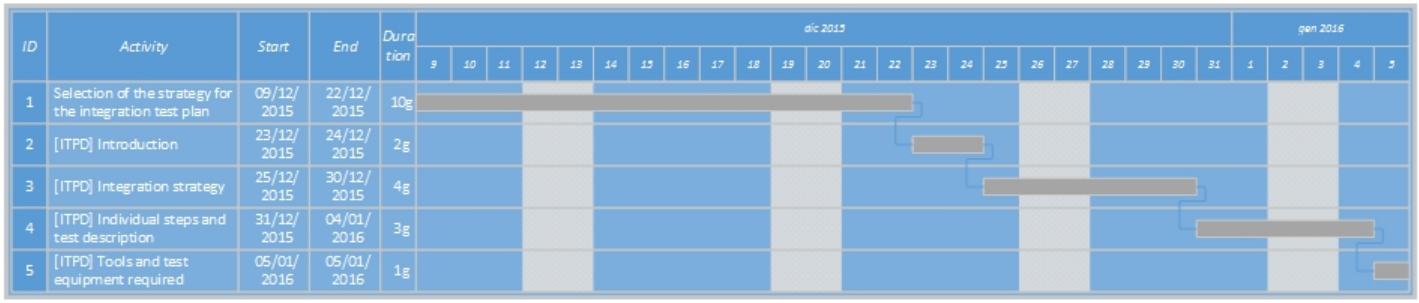
ID	Activity	Start	End	Duration
1	Team creation	12/10/2015	13/10/2015	2g
2	Overview of the application's features	14/10/2015	16/10/2015	3g
3	Overview of the product constraint	19/10/2015	21/10/2015	3g
4	Overview of the specific requirements	22/10/2015	27/10/2015	4g
5	[RASD] Introduction	28/10/2015	28/10/2015	1g
6	[RASD] Overall description	04/11/2015	06/11/2015	3g
7	[RASD] Specific requirements	29/10/2015	03/11/2015	4g
8	[RASD] Alloy	09/11/2015	11/11/2015	3g

3.3.2 DD Scheduling



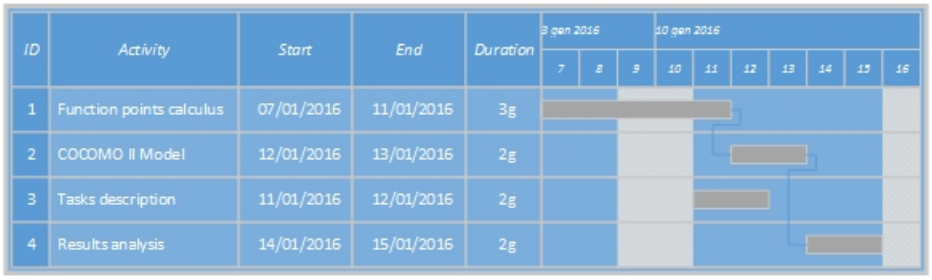
ID	Activity	Start	End	Duration
1	Selection of the architecture design	12/11/2015	19/11/2015	6g
2	[DD] Introduction	20/11/2015	23/11/2015	2g
3	[DD] Architectural design	24/11/2015	26/11/2015	3g
4	[DD] Algorithm design	27/11/2015	02/12/2015	4g
5	[DD] User interface design	03/12/2015	07/12/2015	3g
6	[DD] Requirements traceability	08/12/2015	09/12/2015	2g

3.3.3 Integran Test Plan Scheduling



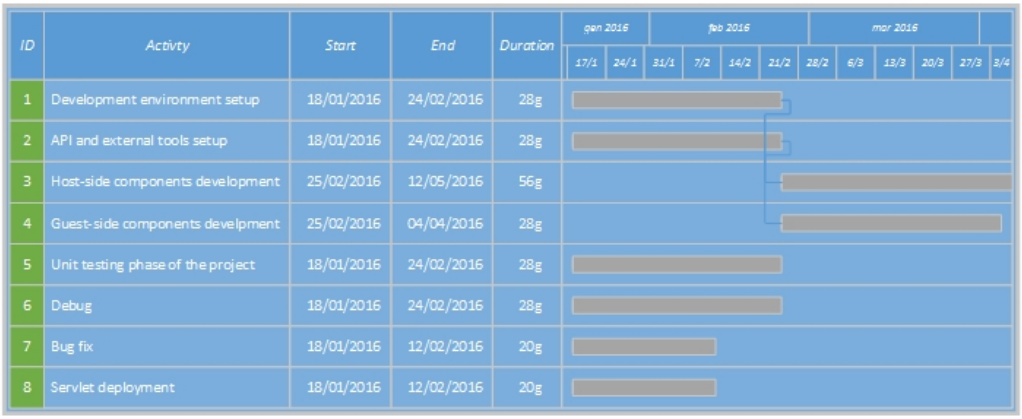
ID	Activity	Start	End	Duration
1	Selection of the strategy for the integration test plan	09/12/2015	22/12/2015	10g
2	[ITPD] Introduction	23/12/2015	24/12/2015	2g
3	[ITPD] Integration strategy	25/12/2015	30/12/2015	4g
4	[ITPD] Individual steps and test description	31/12/2015	04/01/2016	3g
5	[ITPD] Tools and test equipment required	05/01/2016	05/01/2016	1g

3.3.4 Project Planning Scheduling



ID	Activity	Start	End	Duration
1	Function points calculus	07/01/2016	11/01/2016	3g
2	COCOMO II Model	12/01/2016	13/01/2016	2g
3	Tasks description	11/01/2016	12/01/2016	2g
4	Results analysis	14/01/2016	15/01/2016	2g

3.3.5 Development Scheduling



ID	Activty	Start	End	Duration
1	Development environment setup	18/01/2016	24/02/2016	28g
2	API and external tools setup	18/01/2016	24/02/2016	28g
3	Host-side components development	25/02/2016	12/05/2016	56g
4	Guest-side components develpment	25/02/2016	04/04/2016	28g
5	Unit testing phase of the project	18/01/2016	24/02/2016	28g
6	Debug	18/01/2016	24/02/2016	28g
7	Bug fix	18/01/2016	12/02/2016	20g
8	Servlet deployment	18/01/2016	12/02/2016	20g

In this Gantt diagram the final date is cutted in order to keep it clear.

4 Project's risks

4.1 Project Risks

4.1.1 Team's availability

- **Description**: Could happen that the work time estimated for each team's member is more than the real time available
- **Probability**: Moderate
- **Effects**: Not serious
- **Strategy**: If it happens must be chosen between delaying the project ending or engage a new member in the team

4.1.2 Error in the requirements

- **Description**: The requirements thought in the project planning phase contain errors
- **Probability**: Moderate
- **Effects**: Serious
- **Strategy**: Re-write all the documentation (RASD,DD and so forth) and check all the already written code

4.2 Technical Risks

4.2.1 Server dimension

- **Description**: User's set becomes too large and there are many requests in parallel that decrease the system's performance
- **Probability**: High, because of the utility of this application
- **Effects**: Moderate
- **Strategy**: Buy another server, or provide the system with a server that can be easily improved during the life-time of the application

4.2.2 Platform damaged

- **Description**: A generic disaster seriously damage the physical platform on which the server is stated
- **Probability**: Very low
- **Effects**: Catastrophic
- **Strategy**: Could be adopted a cloud solution for the platform or buy more server dislocated in order to avoid this loss of service

4.2.3 Application's bug

- **Description**: A bug in the client platform disables the communication with the server
- **Probability**: Moderate
- **Effects**: Moderate
- **Strategy**: A support team to be called if there are problems in the application

4.3 Business Risks

4.3.1 Possible competitors

- **Description**: New applications rise in this *mobility* field (Car sharing, etc)
- **Probability**: High
- **Effects**: Serious
- **Strategy**: Improve the existing functionality, create new application's features, if the application is not launched yet, move up the releasing date.

5 Appendix

5.1 Software and tool used

In order to redact this document we have used the following softwares:

- – Name: COCOMO tool
  – Version: -
  – Scope: estimate effort, Duration and Cost
  – Reference: `http://csse.usc.edu/tools/COCOMOII.php`

5.2 Working Hours

The Project Plan Document was written in more or less 10 hours divided between the two elements of the group:

- Massimiliano Paci: 5 hours
- Giovanni Patruno: 5 hours

Each element of the group didn't work on specific sections but worked on the whole document