POLITECNICO
DI MILANO

Computer Science and Engineering
A.A. 2016/2017
Software Engineering 2 Project:
Power&Joy

Project Plan

January 22, 20167

Prof. Luca Mottola
Joshua Nicolay Ortiz Osorio Matr: 806568
Michelangelo Medori Matr: 878025

Index

1 Introduction

1.1 Revision History

Version	Date	Author(s)	Summary
1.0	22/01/2017	Joshua Nicolay Ortiz Osorio and Michelangelo Medori	initial release
2.0	07/02/2017	Joshua Nicolay Ortiz Osorio and Michelangelo Medori	added park data to FP

1.2 Purpose and Scope

This is the Project Plan Document for the Power&Joy car sharing management system. The aim of this document is to provide a detailed estimation of the size and the cost, given by an accurate analysis of the complexity of the functionalitires of the system. After the following list of definitions, acronyms and abbreviations and reference documents, in section 2 we are going provide first a size estimation using the Function Points approach (chapter 2.1) and a cost/effort estimation using the COCOMO model. In section 3 we are going to describe a possible schedule for the project development, which covers all the activities involved in the development, ranging from the requirements analysis to the implementation and testing activities. In section 4 we are going to assign to each member of our development group various tasks in order to allocate the right resources to every task. Finally, section 5 contains an analysis of the risks that arise about the project development, and some overall conclusions.

1.3 List of Definitions, Acronyms, Abbreviations

1.3.1 Definitions

1.3.2 Acronyms

- FP: Function Points
- ILF: Internal Logic File
- ELF: External Logic File
- EI: External Input
- EO: External Inquiry
- EQ: External Output
- DBMS: Database Management System.
- UI: User Interface.
- GPS: Global Positioning System.
- TDI : Total Degree of Influence
- VAF: Value Adjustment Factor
- UFP: Unadjusted Function Point
- AFP : Adjusted Function Point
- SLOC: Source Lines Df Code

1.3.3 Abbreviations

1.4 Reference Documents

- Assignment document.pdf
- RASD (Requirement Analysis and Specification Document 4.0.pdf)
- Project Plan Example Document.pdf
- The Function Points complexity evaluation tables
- The COCOMO II Model Definition Manual (version 2.1, 1995 ? 2000 Center for Software Engineering, USC).

2 Project size, cost and effort estimation

in this section we provide first a size estimation (chapter 2.1) using mainly the Function Points approach, which is apt to estimate the number of lines of code needed to develop the Power&Joy car sharing management system, analysing all its main functionalities and providing a size estimation for each of them. We are going to focus only on the business logic implementation, leaving apart the development of the user interface. in section 2.2 we provide an effort and cost estimation using the COCOMO model and the size estimation found in 2.1.

2.1 Size estimation: function points

The function points approach provides an estimation of the size of the project in terms of lines of code taking as inputs the functionalities that have to be performed by the system and assigning to them values that represent their complexity. The complexity is computed according to the following tables, that divide the functionalities in Internal Logic Files (ILFs), External Logic Files(ELFs), External Inputs(EIs), External Inquiries(EQs) and External Outputs(EOs).

ILFs and ELFs

ILF/EIF	1-19 DET	20-50 DET	50+ DET
1 FTR	Low	Low	Average
2-5 FTR	Low	Average	High
6 or more FTR	Average	High	High

EOs and EQs

EO/EQ	1-5 DET	6-19 DET	20+ DET
0-1 FTR	Low	Low	Average
2-3 FTR	Low	Average	High
4 or more FTR	Average	High	High

EIs

EI	1-4 DET	5-15 DET	16+ DET
0-1 FTR	Low	Low	Average
2 FTR	Low	Average	High
3-4 or more FTR	Average	High	High

UFP complexity weights

UFP	WEIGHT		
Function Types	Simple	Medium	Complex
External Input (EI)	3	4	6
External Queries (EQ)	3	4	6
External Output (EO)	4	5	7
External Interface File (EIF)	5	7	10
Internal Logic File (ILF)	7	10	15

AFP complexity weights

Number	Complexity Weighting Factor
1	Backup and recovery
2	Data communication
3	Distributed processing
4	Performance critical
5	Existing operating environment
6	On-line data entry
7	Input transaction over multiples screens
8	Master files updated online
9	Information domain values complex
10	Internal processing complex
11	Code designed for reuse
12	Conversion/installation in design
13	Multiple installations
14	Application designed for change

Value	Characteristic
0	Not Present, No Influence
1	Incidental Influence
2	Moderate Influence
3	Average Influence
4	Significant Influence
5	Strong Influence throughout

2.1.1 InternalLogicFiles(ILFs)

In this paragraph we identify and analyse the IFls (i.e the internal logic files used by the application in order to store the data it needs to provide its functionalities) .

User Data The application uses a single table to store all the data about the users, that consist of

first name : String

last name : String

userId : String

email : String

password : String

fiscal Code : String

driving license code : String

credit card number code : String

birthday date : Date

Operator Data The application uses a single table to store all the data about the operators, that consist of:

first name : String

last name : String

operatorId : String

email : String

password : String

fiscal Code : String

driving license code : String

stationId: String

birthday date : Date

Car Data
The application uses a single table to store all the data about the electric cars. The table contains the following set of attributes :

carId : String

userId : String

currentBattery : float

latitude : float

longitude : float

issues : String

in repair : Boolean

isAvailable: Boolean

Station Data
The application uses a single table to store information about the stations:

stationId : String

latitude : float

longitude : float

cars : int

Reservation Data
For the reservation data the application uses 2 different tables: one is used to store the ongoing reservations (i.e. those which have been done by users within an hour from the current time, and haven't been canceled), while the other table is used to store all past reservations, including the ones that have been cancelled. Both tables have the following set of attributes:

reservationId : String

userId : String

carId : String

date : Date

timer : Float

isCancelled : Boolean

Ride Data

For the ride data the application uses 2 different tables: one is used to store the ongoing rides (i.e. those which have been started by users and haven't ended yet), while the other table is used to store all past rides. Both tables have the following set of attributes:

rideId : String

userId : String

carId : String

date : Date

initialTime : Float

finalTime : Float

initialPosition : Float

finalPosition: Float

passengers : Int

pitstops : Int

isOnCharge : Boolean

finalPrice : Float

pitstopTotalLength : float

totalRideLength : Int

lowCharge : Boolean

accident: Boolean

Given the information above, w assign to each ILF the following scale factors.

ILF	Complexity	FPs
User Data	Low	7
Operator Data	Low	7
Car Data	Low	7
Station Data	Low	7
Ride Data	Low	7
Reservation Data	Low	7
Car Park	Low	7
Total		49

2.1.2 External Logic Files (ELFs)

The main external source of data of the application comes from the mapping service, which is mainly used to perform these operations:

- Given a starting and a final locations, compute the shortest path (as a sequence of locations) from the first location to the destination
- Given an address, get the correspondent pair of coordinates (reverse geocoding)

On the client side, the mapping service is also used to retrieve the graphical representation of the city map to be displayed on the user's home page .
The following table shows the assignments of the FPs values to each one of the ELF above:

ELF	Complexity	FPs
Shortest Path Computation	Low	5
Reverse Geocoding	Low	5
Map Data Retrieval	Low	5
Total		15

2.1.3 External Inputs (EIs)

The Power&Joy application supports many kinds of interactions with users and operators.
Users:

- **Login/Logout (User)**these are simple operations, and count for 3FPs each
- **Make reservation** This is a complex operation, so we are giving it 6 FPs
- **Cancel reservation** This is a simple operation valuable of 3 FPs
- **Start/End ride** Thse are complex operations, and require 6 FPs each
- **Start/End PitStop** These are complex operations and require 4FPs each
- **Report Accident** This is a complex operation, and requres 4FPs
- **Assistance Needed** This is a simple operation, and involves only 3FPs
- **Set User profile** This is a complex operation and rquires 4FPs
- **Password retreival** This is a complex operation, and requires 4FPs

- **View historical rides** This is a simple operation, and requires only 3FPs
- **User Registration** This is a complex operation, and requires 4FPs

Operators:

- **operator Registration** This is a complex operation, and requires 4 FPs
- **Login/Logout operator** These are simple operations, and require 3FPs each
- **Operate**This is a complex operation, and require 6FPs
- **Issue Solved** This is a quite complex operation, valuable for 4Fps
- **Request Service Statistics**These operation involves complex queries, so it requires 6FPs

The following table assigns the FPs to each one of these functions:

EI	Complexity	FPs
Login/Logout (User)	Low	2*3
Make Reservation	High	6
Cancel Reservation	Low	3
Start/End Ride	High	2*6
Start/End Pit-Stop	Low	2*3
Report Accident	Low	3
Assistance Needed	Low	3
Password Retrieval	Average	4
Set User Profile	Average	4
View Historical Rides	Low	3
User Registration	Average	4
Operator Registration	Average	4
Login/Logout (Operator)	Low	2*3
Operate	Low	3
Issue Solved	Low	3
Request Service Statistic	High	6
Total		76

2.1.4 External Inquiries (EQs)

Since inquiries are just data requests from the users and operators, they can be seen as simple operations. The inquiries supported by our application are the following:

Users:

- Request cars to reserve
- Retrieve reservation info
- Retrieve ride info
- Retrieve personal profile data
- Retrieve reservation history

Operators:

- Request the list of all cars
- request the list of issued cars
- request the list of cars the are being repaired
- Retrieve reservation info
- Retrieve ride info
- Retrieve personal profile data
- Retrieve reservation history

The following table assigns the FPs to each one of these functions:

EQ	Complexity	FPs
Retrieve cars to reserve	Low	3
Retrieve reservation info	Low	3
Retrieve Ride info	Low	3
Retrieve personal profile	Low	3
Retrieve Reservation history	Low	3
Retrieve list of cars	Low	3
Retrieve list of issued cars	Low	3
Retrieve list of reservations	Low	3
Retrieve list of rides	Low	3
Retrieve list of cars in repair	Low	3
Total		30

2.1.5 External Outputs (EOs)

Th Power&Joy application sometimes needs to communicate with users and operators outside the contest of an enquiry, in order to accomplish the following functions:
Users

- Notify user that car has been reserved
- Notify user the cost of the ride when it end
- Notify the user that he left the car 3km (or more) away from the nearest charging station (at the end of the ride)

Operators

- Notify the operator with the information (current position, issues, user, ecc.) about the car he wants to take car of

Moreover, the system also needs to communicate with the external system that takes care of the payments for the rides.
The following table assigns the FPs to each one of these functions:

EO	Complexity	FPs
Reservation notification	Low	4
Cost notification	Low	4
3 km away notification	Low	4
Car info notification	Low	4
Payment Data	Low	4
Total		20

2.1.6 Overall estimation

Before going on with the final results, we are going to calculate the adjustment factors in order to compute the TDI (Total degree of influence) :

Number	Complexity Weighting Factor	Values
1	Backup and recovery	3
2	Data communication	3
3	Distributed processing	2
4	Performance critical	4
5	Existing operating environment	0
6	On-line data entry	3
7	Input transaction over multiples screens	2
8	Master files updated online	4
9	Information domain values complex	2
10	Internal processing complex	3
11	Code designed for reuse	4
12	Conversion/installation in design	2
13	Multiple installations	2
14	Application designed for change	3
Total		37

Thus we have

$$TDI = 37$$

so that the VAF (Value adjustment factor) results

$$VAF = (TDI/100) + 0.65 = (37/100) + 0.65 = 1.02$$

The following table summarises the results of our EFP estimation activity:

Function type	Values
Internal Logic Files External Logic Files External Inputs External Inquiries External Outputs	49 15 76 30 20
Total	190

Considering Java Enterprise Edition as a development platform and disregarding the aspects concerning the implementation of the mobile applications (which can be thought as pure presentation with no business logic), we can estimate the total number of lines of code. Depending on the conversion rate, we have a lower bound of:

$$UFP = 190 * 46 = 8740$$

and an upper bound of

$$UFP = 190 * 67 = 12730$$

and the adjusted values are computed as

$$SLOC = UFP * VAF$$

which gives as an upper bound

$$SLOC = 8740 * 1.02 = 8914$$

and as a lower bound

$$SLOC = 12730 * 1.02 = 12985$$

2.2 Cost and effort estimation: COCOMO II

In this section we're going to use the COCOMO II approach to estimate the cost and effort needed to develop the Power&Joy system .

2.2.1 Scale Drivers

In order to evaluate the values of the scale drivers, we refer to the following o COCOMO II table

Scale Factor values, SF_j , for COCOMO II Models

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC SF_j	Thoroughly unprecedented **6.20**	Largely unprecedented **4.96**	Somewhat unprecedented **3.72**	General familiar **2.48**	Largely familiar **1.24**	Thoroughly familiar **0.00**
FLEX SF_j	Rigorous **5.07**	Occasional relaxation **4.05**	Some relaxation **3.04**	General conformity **2.03**	Some conformity **1.01**	General goals **0.00**
RESL SF_j	Little (20%) **7.07**	Some (40%) **5.65**	Often (60%) **4.24**	Generally (75%) **2.83**	Mostly (90%) **1.41**	Full (100%) **0.00**
TEAM SF_j	Very difficult interactions **5.48**	Some difficults interactions **4.38**	Basically cooperative interactions **3.29**	Largely cooperative **2.19**	Highly cooperative **1.10**	Seamless interactions **0.00**
PMAT SF_j	Level 0 Lower **7.80**	Level 1 Upper **6.24**	Level 2 **4.68**	Level 3 **3.12**	Level 4 **1.56**	Level 5 **0.00**

Follows a short description for each driver:

Precedenteness it's a parameter that describes the previous experience of our team with the development of large scale projects. Since we are not expert in the field, this value will be low.

Development flexibility it's a parameter that describes the degree of flexibility in the development process with respect to the external specification and requirements. Since there are very strict requirements on the functionalities but nothing specific is stated as for the technology to be used, this value will be low.

Risk resolution it's a parameter that describes the level of awareness and reactiveness with respect to risks. The risk analysis we performed is quite extensive, so the value will be set to high.

Team cohesion it's a parameter that describes how well the team members know each other and work together in a cooperative way. For our team, the value is very high.

Process maturity it's a parameter that describes how well the goals have been achieved. Since we managed to achieve every goal, even if with some difficulties, this parameter will be medium valuated

These are the final results:

Scale Driver	Factor	Value
Precedentedness (PREC)	Low	4.96
Development flexibility (FLEX)	Low	4.05
Risk resolution (RESL)	High	2.83
Team cohesion (TEAM)	Very High	1.10
Process maturity (PMAT)	Normal	4.68
Total		17.62

2.2.2 Cost Drivers

These are the cost drivers we are going to refer to in order to estimate the cost of the development of our application:

- Required Software Reliability: Since the application is the only way to reserve and drive the electric cars, the RELY cost driver is set to high.

Required software reliability

RELY Cost Drivers						
RELY Descriptors	Slightly inconvenience	Easily recoverable losses	Moderate recoverable losses	High financial loss	Risk human life	
Rating level	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: Since the SLOC value is ranging between 8000 and 12000, we decided to place the DATA driver to normal .

Database size

DATA Cost Drivers						
DATA Descriptors					>1000	
Rating level	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: is set to high, given the results of the scale drivers .

Product complexity

CPLX Cost Driver						
Rating level	Very Low	Low	Nominal	High	Very High	Extra High
Effort multipliers	0.73	0.83	1.00	1.17	1.34	1.74

- Required reusability: reusability is really important in our project, for further developments plans, so the RUSE driver is set to high.

Required reusability

RUSE Cost Driver						
RUSE Descriptors		None	Across project	Across program	Across product line	Across multiple product lines
Rating level	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: set to normal, since every aspect of our application has been analysed in the documentation.

Documentation reausability

DOCU Cost Driver						
DOCU Descriptors	Many life-cycle needs uncovered	Some life-cycle needs uncovered	Right-size to lifecycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
Rating level	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: We expect the CPU usage is going to be very high, given the project complexity and the expected number of users.

Execution time constraint

TIME Cost Driver						
TIME Descriptors			use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating level	Very Low	Low	Nominal	High	Very High	Extra High
Effort multipliers	n/a	n/a	1.00	1.11	1.29	1.63

- Storage constraint: this parameter is set to normal, since we don't expect having problems of storage, given the currently available technologies.

Storage constraint

STOR Cost Driver						
STOR Descriptors			use of available storage	70% use of available storage	85% use of available storage	95% use of available storage
Rating level	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: this parameter is set to normal, since we don't expect to be doing major updates really often, except for the clients.

Platform volatility

PVOL Cost Driver						
PVOL Descriptors		Major change every 12 months, minor change every 1 months	Major: 6 months ; minor: 2 weeks	Major: 2 months; minor: 1 week	Major: 2 weeks; minor: 2 days	
Rating level	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: We think the analysis of the problem has been conducted in a thorough and complete way with respect to a potential real world implementation. For this reason, this parameter is set to high.

Analist Capability

ACAP Cost Driver						
ACAP Descriptors	15^{th} percentile	35^{th} percentile	55^{th} percentile	75^{th} percentile	90^{th} percentile	
Rating level	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: Set to normal, since we don't have that much of experience in project implementation yet.

Programmer Capability

PCAP Cost Driver						
PCAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating level	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: we have no much experience in the development of such large scale applications, so we set this driver to low.

Application Experience

APEX Cost Driver						
APEX Descriptors	2 months	6 months	1 year	3 years	6 years	
Rating level	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: We don't have any experience with the Java EE platform, so this parameter is set to normal

Platform Experience

PLEX Cost Driver						
PLEX Descriptors	2 months	6 months	1 year	3 years	6 years	
Rating level	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: We don't have any experience with the Java EE platform, so this parameter is set to normal

Languages and Tool Experience

LTEX Cost Driver						
LTEX Descriptors	2 months	6 months	1 year	3 years	6 years	
Rating level	Very Low	Low	Nominal	High	Very High	Extra High
Effort multipliers	1.20	1.09	1.00	0.91	0.84	n/a

- Personnel continuity: set to very low, since we don't have to actually develop the software

Personnel continuity

PCON Cost Driver						
PCON Descriptors	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating level	Very Low	Low	Nominal	High	Very High	Extra High
Effort multipliers	1.29	1.12	1.00	0.90	0.81	n/a

- Usage of Software Tools: Our application environment is complete and well integrated, so we'll set this parameter to high.

Usage of software Tools

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, moderated integrated	Strong mature, proactive life-cycle tools, wells integrated with processes, methods, reuse	
Rating level	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: we mostly communicated via Internet while working on this project, so we are going to set this parameter to high

Multisite development

SITE Cost Driver						
SITE Collocation Descriptors SITE Communication Descriptors	International Some phone, mail	Multi-city and multi-company Individual phone, fax	Multi-city or multi-company Narrow band email	Same city or metro area Wideband electronic communication	Same building or complex Wideband elect. comm. , occasional video conf.	Fully collocated interactive multimedia
Rating level	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: The definition of all the required documentation took a consistent amount of time, especially for the requirement analysis and the design stages. For this reason, this parameter is set to high.

Required development schedule

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 multipliers	1.43	1.14	1.00	1.00	1.00	n/a

The following table reassumes the value assignments for each one of the fields described above:

Cost Driver	Factor	Value
Required Software Relaiability (RELY)	High	1.10
Database size (DATA)	Normal	1.00
Product complexity (CPLX)	High	1.17
Required Reusability (RUSE)	High	1.07
Documentation match to cycle-needs (DOCU)	Normal	1.00
Execution Time Constraint (TIME)	Very High	1.29
Main Storage constraint (STOR)	Normal	1.00
Platform volatility (PVOL)	Normal	1.00
Analyst capability (ACAP)	High	0.85
Programmer capability (PCAP)	Normal	1.00
Application Experience (APEX)	Low	1.10
Platform Experience (PLEX)	Normal	1.00
Language and Tool Experience (LTEX)	Normal	1.00
Personnel continuity (PCON)	Very Low	1.29
Usage of Software Tools (TOOL)	High	0.90
Multisite development (SITE)	High	0.93
Required development schedule (SCED)	High	1.00
Total		1.79

2.2.3 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 II)
EAF=product of all cost drivers (1.79)
E = exponent derived from the scale drivers. It is computed as:

$$E = B + 0.01 * \sum SF[i] = B + 0.01 * 17.62 = 0.91 + 0.1762 = 1.0862$$

in which B is equal to: 0.91 for COCOMO II.
With this parameters we can compute the effort value, which has a lower bound of:

$$Effort = A * EAF * KSLOC^E = 2.94 * 1.79 * 8.740^{1.0862} = 55.44PM = 56PM$$

and a lower bound of:

$$Effort = A * EAF * KSLOC^E = 2.94 * 1.79 * 12.985^{1.0862} = 85.23PM = 86PM$$

2.2.4 Schedule estimation

As concerns the final schedule, we are going to use the following formula:

$$Duration = 3.67 * Effort^E$$

where F is computed as :

$$F = 0.28 + 0.2 * E - B = 0.28 + 0.2 * 1.0862 - 0.91 = 0.31524$$

having E = 1.0862 and B = 0.91 as seen above
Finally As a lower bound, we have Effort = 54.39 and

$$Duration = 3.67 * 55.44^{0.31524} = 13.0133 months$$

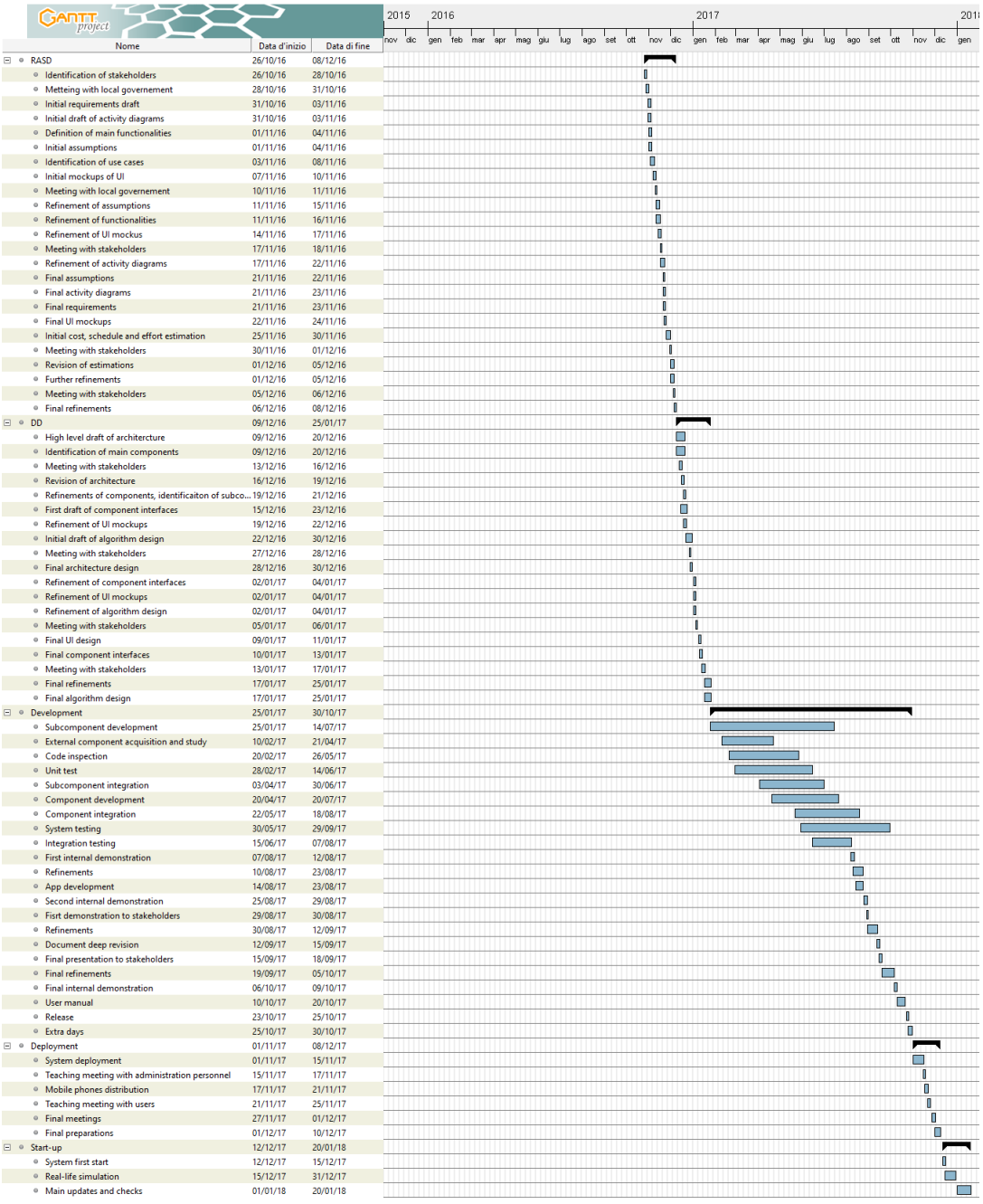
and as an upper bound we have Effort = 81.82 and

$$Duration = 3.67 * 85.23^{0.31524} = 14.9027 months$$

which both seem reasonable results.

3 Schedule

In this chapter we're going to provide an high-level project schedule. More refined schedules will be defined during the project to manage the internal organisation of the single development phases. Notice that, even if this project is being for didactic purposes (such that no implementation and testing activities are going to be performed), we have considered these steps as part of our schedule as well. In order to take into account what could be the full development of this project.

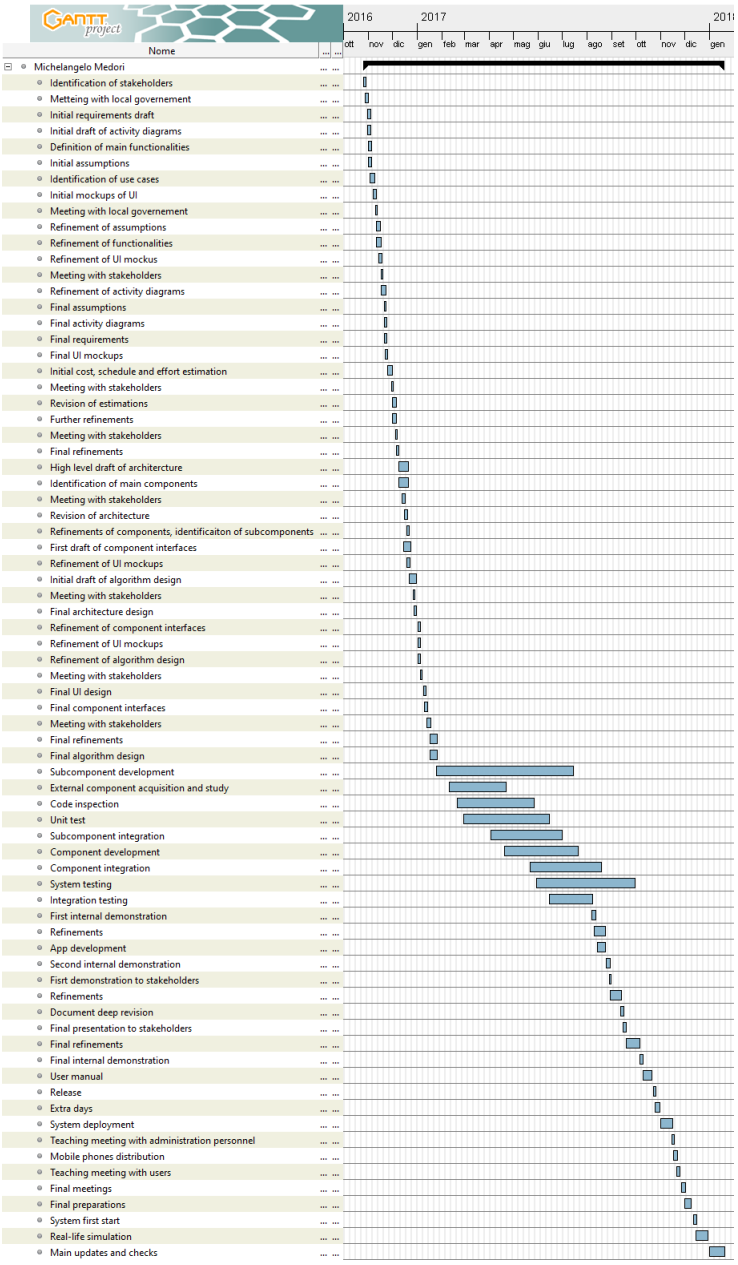


Nome	Data d'inizio	Data di fine
RASD	26/10/16	08/12/16
Identification of stakeholders	26/10/16	28/10/16
Metteing with local governement	28/10/16	31/10/16
Initial requirements draft	31/10/16	03/11/16
Initial draft of activity diagrams	31/10/16	03/11/16
Definition of main functionalities	01/11/16	04/11/16
Initial assumptions	01/11/16	04/11/16
Identification of use cases	03/11/16	08/11/16
Initial mockups of UI	07/11/16	10/11/16
Meeting with local governement	10/11/16	11/11/16
Refinement of assumptions	11/11/16	15/11/16
Refinement of functionalities	11/11/16	16/11/16
Refinement of UI mockus	14/11/16	17/11/16
Meeting with stakeholders	17/11/16	18/11/16
Refinement of activity diagrams	17/11/16	22/11/16
Final assumptions	21/11/16	22/11/16
Final activity diagrams	21/11/16	23/11/16
Final requirements	21/11/16	23/11/16
Final UI mockups	22/11/16	24/11/16
Initial cost, schedule and effort estimation	25/11/16	30/11/16
Meeting with stakeholders	30/11/16	01/12/16
Revision of estimations	01/12/16	05/12/16
Further refinements	01/12/16	05/12/16
Meeting with stakeholders	05/12/16	06/12/16
Final refinements	06/12/16	08/12/16
DD	09/12/16	25/01/17
High level draft of architercture	09/12/16	20/12/16
Identification of main components	09/12/16	20/12/16
Meeting with stakeholders	13/12/16	16/12/16
Revision of architecture	16/12/16	19/12/16
Refinements of components, identificaiton of subco...	19/12/16	21/12/16
First draft of component interfaces	15/12/16	23/12/16
Refinement of UI mockups	19/12/16	22/12/16
Initial draft of algorithm design	22/12/16	30/12/16
Meeting with stakeholders	27/12/16	28/12/16
Final architecture design	28/12/16	30/12/16
Refinement of component interfaces	02/01/17	04/01/17
Refinement of UI mockups	02/01/17	04/01/17
Refinement of algorithm design	02/01/17	04/01/17
Meeting with stakeholders	05/01/17	06/01/17
Final UI design	09/01/17	11/01/17
Final component interfaces	10/01/17	13/01/17
Meeting with stakeholders	13/01/17	17/01/17
Final refinements	17/01/17	25/01/17
Final algorithm design	17/01/17	25/01/17
Development	25/01/17	30/10/17
Subcomponent development	25/01/17	14/07/17
External component acquisition and study	10/02/17	21/04/17
Code inspection	20/02/17	26/05/17
Unit test	28/02/17	14/06/17
Subcomponent integration	03/04/17	30/06/17
Component development	20/04/17	20/07/17
Component integration	22/05/17	18/08/17
System testing	30/05/17	29/09/17
Integration testing	15/06/17	07/08/17
First internal demonstration	07/08/17	12/08/17
Refinements	10/08/17	23/08/17
App development	14/08/17	23/08/17
Second internal demonstration	25/08/17	29/08/17
Fisrt demonstration to stakeholders	29/08/17	30/08/17
Refinements	30/08/17	12/09/17
Document deep revision	12/09/17	15/09/17
Final presentation to stakeholders	15/09/17	18/09/17
Final refinements	19/09/17	05/10/17
Final internal demonstration	06/10/17	09/10/17
User manual	10/10/17	20/10/17
Release	23/10/17	25/10/17
Extra days	25/10/17	30/10/17
Deployment	01/11/17	08/12/17
System deployment	01/11/17	15/11/17
Teaching meeting with administration personnel	15/11/17	17/11/17
Mobile phones distribution	17/11/17	21/11/17
Teaching meeting with users	21/11/17	25/11/17
Final meetings	27/11/17	01/12/17
Final preparations	01/12/17	10/12/17
Start-up	12/12/17	20/01/18
System first start	12/12/17	15/12/17
Real-life simulation	15/12/17	31/12/17
Main updates and checks	01/01/18	20/01/18

4 Resource Allocation

In this chapter we're going to provide a general overview of how the tasks defined by the schedule in the previous section will be divided between the two members of our development team. More refined schedules will be defined during the project to manage the internal organisation of the single development phases. As we already mentioned in the previous section, we have also included activities in the requirement analysis and design phases that won't actually take place, like the stakeholders meetings, as well as the full implementation phase. This has been purposefully done to have a more realistic depiction of how the development process could go.


GANTT project
2016
2017
ott nov dic gen feb mar apr mag giu lug ago set ott nov dic gen
Nome
Joshua Nicolay Ortiz Osorio
Identification of stakeholders
Metteing with local governement
Initial requirements draft
Initial draft of activity diagrams
Definition of main functionalities
Initial assumptions
Identification of use cases
Initial mockups of UI
Meeting with local governement
Refinement of assumptions
Refinement of functionalities
Refinement of UI mockus
Meeting with stakeholders
Refinement of activity diagrams
Final assumptions
Final activity diagrams
Final requirements
Final UI mockups
Initial cost, schedule and effort estimation
Meeting with stakeholders
Revision of estimations
Further refinements
Meeting with stakeholders
Final refinements
High level draft of architercture
Identification of main components
Meeting with stakeholders
Revision of architecture
Refinements of components, identificaiton of subcomponents
First draft of component interfaces
Refinement of UI mockups
Initial draft of algorithm design
Meeting with stakeholders
Final architecture design
Refinement of component interfaces
Refinement of UI mockups
Refinement of algorithm design
Meeting with stakeholders
Final UI design
Final component interfaces
Meeting with stakeholders
Final refinements
Final algorithm design
Subcomponent development
External component acquisition and study
Code inspection
Unit test
Subcomponent integration
Component development
Component integration
System testing
Integration testing
First internal demonstration
Refinements
App development
Second internal demonstration
Fisrt demonstration to stakeholders
Refinements
Document deep revision
Final presentation to stakeholders
Final refinements
Final internal demonstration
User manual
Release
Extra days
System deployment
Teaching meeting with administration personnel
Mobile phones distribution
Teaching meeting with users
Final meetings
Final preparations
System first start
Real-life simulation
Main updates and checks


GANTT project
2016
2017
2018
ott nov dic gen feb mar apr mag giu lug ago set ott nov dic gen
Nome
Michelangelo Medori
Identification of stakeholders
Metteing with local governement
Initial requirements draft
Initial draft of activity diagrams
Definition of main functionalities
Initial assumptions
Identification of use cases
Initial mockups of UI
Meeting with local governement
Refinement of assumptions
Refinement of functionalities
Refinement of UI mockus
Meeting with stakeholders
Refinement of activity diagrams
Final assumptions
Final activity diagrams
Final requirements
Final UI mockups
Initial cost, schedule and effort estimation
Meeting with stakeholders
Revision of estimations
Further refinements
Meeting with stakeholders
Final refinements
High level draft of architercture
Identification of main components
Meeting with stakeholders
Revision of architecture
Refinements of components, identificaiton of subcomponents
First draft of component interfaces
Refinement of UI mockups
Initial draft of algorithm design
Meeting with stakeholders
Final architecture design
Refinement of component interfaces
Refinement of UI mockups
Refinement of algorithm design
Meeting with stakeholders
Final UI design
Final component interfaces
Meeting with stakeholders
Final refinements
Final algorithm design
Subcomponent development
External component acquisition and study
Code inspection
Unit test
Subcomponent integration
Component development
Component integration
System testing
Integration testing
First internal demonstration
Refinements
App development
Second internal demonstration
Fisrt demonstration to stakeholders
Refinements
Document deep revision
Final presentation to stakeholders
Final refinements
Final internal demonstration
User manual
Release
Extra days
System deployment
Teaching meeting with administration personnel
Mobile phones distribution
Teaching meeting with users
Final meetings
Final preparations
System first start
Real-life simulation
Main updates and checks

5 Risk Management

In this section we are going to analyse the main risks that the project development may face. Some of them could derive from possible technical issues, while others come from political or financial dynamics . First of all, a major threat is represented by our main political stakeholder, which is the city administration. Notice that up to now we have been considering Milan's urban centre, but without focusing or relying much to Italy's laws, since we assumed that the system could be applied to any industrialised city, or even expand from a single city to an entire region or nation. Potential issues can include a change of the city government, a budget crisis (possibly as a secondary effect of some nationwide policy of spending review) or other shifts in the local government priorities. The main solution to this issue is to actively involve the political stakeholders (i.e. at least member of the city government) in the requirement analysis and design stages, as well as the implementation one. Activities in this direction may include periodical reviews and meetings, demonstrations, discussions on the interface design and so on, with possible negotiations to be made. Another related political issue concerns the possible changes in the national legislation. In particular, we are mainly concerned with modifications to the way car sharing services are operated in general (for instance revisions of the driving license format or other restrictions), on the rules concerning electric cars (for example charges on the electric current or on the disposal of the charging stations) and to the possibility of using a smartphone while driving. There are already some limitations to this, but today they are overcome by avoiding to actively interact with the device and by keeping it fixed in a position that doesn?t interfere with the vision of the road. Stricter laws could be enacted in the future that forbid this possibility and require, for instance, that only voice interactions are allowed. Other issues might arise regarding the acceptance of the system from its intended users, both taxi drivers and passengers. As this system is going to completely replace the previous taxi management service, it is reasonable to assume that it?s going to face some initial opposition from people unwilling to change their habits (maybe because of the electric cars concept). To make the transition easier, we suggest considering a few marketing strategies aimed at winning the support of the majority of the users. These can include acceptance tests, special offers and discounts for an initial period of time or other kinds of incentives. We also have to consider issues arising from people management inside our company. Key members of the team may get ill just prior to important milestones or meetings, or may be ill for prolonged periods of time, causing delays. Also, we have to consider the possibility of people quitting the company, as the IT job market is quite flexible. A possible solution for this problem is to split duties and responsibilities across multiple people, so that no single person is in charge of a specific task. Another risk might come from overestimating the knowledge of a specific matter or programming technique that our programmers and engineers have. Adding people to the project should not be seen as the primary solution here, unless the task is extremely

specific. A good antidote is to hire knowledgable and flexible people beforehand. Obviously, a loss of the whole source code, or significant parts of it , would be a disaster. This issues is quite easy to prevent, by implementing appropriate versioning systems and backup techniques distributed over multiple, redundant locations. Another issue that must not be underestimated is related to our dependency on external services and components. A change in the terms and conditions of the Mapping Service, for example, could pose serious financial or technical problems. We are somewhat more protected as for database as there is a greater number of vendors and the access methods are more or less standardised. Also, a change in the pricing plans of the cloud infrastructure could lead to significant issues on the financial and business side. A final problem may also emerge from issues with the project scheduling. Even though an initial overall schedule is provided in this document, it can?t obviously take into account all the possible issues that may arise down the road. For this reason, some extra time has been allocated at the expected end of each major activity to allow for adjustments.

6 ChangeLog

- Version 1.0 : initial version

7 Hours of Work

To redact this document, we spent 15 hours per person. We also report here the overall amount of hours per person required by the project.