POLITECNICO
MILANO 1863

Computer Science and Engineering
Project of Software Engineering 2

PowerEnJoy

Project Planning Document

Authors:

Alberto Zeni 813977

Manuel Parenti 876085

Reference Professor: Mottola Luca

Academic Year 2016–2017

Table of Contents

1 Introduction **3**
1.1 Changelog **3**
1.2 Purpose and Scope **3**
1.3 Definitions, Acronyms, Abbreviations **3**
1.3.1 Definitions 3
1.3.2 Acronyms 4
1.4 Reference Documents **4**
2 Project size, cost and effort estimation **5**
2.1 Size estimation: function points **5**
2.1.1 Internal Logic Files (ILs) 6
2.1.2 External Logic Files (ELs) 8
2.1.3 External Inputs (EIs) 8
2.1.4 External Inquiries (EQs) 10
2.1.5 External Outputs (EOs) 12
2.1.6 Overall Estimation 13
2.2 Cost and effort estimation: COCOMO II **13**
2.2.1 Scale Factors 13
2.2.2 Cost Drivers 15
2.2.3 Effort equation 22
2.2.4 Schedule estimation 23
3 Schedule **23**
4 Resource Allocation **25**
5 Risk Management **27**
6 Additional Information **28**
6.1 Hours of Work **28**
6.2 Used Tools **28**

1 Introduction

1.1 Changelog

V1.0 First Release
V1.1 Text Fixes (some formulas were not displayed correctly)

1.2 Purpose and Scope

This document represents the Plan Document for our project, PowerEnJoy. The main purpose of this document is to give some estimations about the project in terms of complexity, resources and effort needed in order to develop the PowerEnJoy system. This information provided by the document can be subsequently used as a reference for the project manager in order to schedule tasks for the stakeholders, manage the resources and define the project budget. The document is divided in four main sections. The first one describes the estimation of the dimension of the project, using the Function Points Approach, and then correlates these data with cost and effort relying on the COCOMO approach. The second section of the document proposes a possible schedule for the project from the requirement analysis to the testing activities using the previously explained estimations. In the third paragraph of the document we have assigned the various tasks of the development to the various stakeholders. In the last part we have examined the possible risks that our project could encounter during the various phases of development and we have provided some thoughts in order to avoid them or, if we cannot do anything else, handle them in a proper and organized way.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

SUBSYSTEM: functional unit observed at high level of the system.

SUBCOMPONENT: each of the components that provides functionalities of the subsystem.

USER: A person who is registered on PowerEnJoy, he can register, find information about his account, search for a car and use the car sharing service using a web application via a browser or using a mobile application. Users are provided with an account ID and a password which they need to use to access the service. Sometimes in the document they are referred to as Customers.

SAFE AREA: Public parking areas defined by the PowerEnJoy company, these won’t cause any police officer to write a ticket for the car or even remove the car.

SPECIAL PARKING AREA: These areas are defined by the PowerEnJoy company and they are provided with electric plugs for charging the battery of the cars.

RESERVATION: This term refers to the possibility of reserving a car for a specific user. He can do this with the web application or the mobile app, after he finds a car he wants to pick up. The cars will be reserved for that user for an hour, and after the time expires if the user doesn't pick up the car he is charged of a small fee. Users can reserve a car for up to an hour before they need to drive.

DISPATCHER: A person or a system that will manage the parking and recharging of the cars. The dispatcher will send operators on field to do the operations required in order to overcome the eventual bad behavior of some users.

GPS: The Global Positioning System is a global navigation satellite system that provides geolocation and time information to a GPS receiver in all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites.

SENSOR: A sensor is a device that converts real world data (Analog) into data that a computer can understand.

1.3.2 Acronyms

FP: Function Points.

ILF: Internal logic file.

ELF: External logic file.

EI: External Input.

EO: External Output.

EQ: External Inquiries.

1.4 Reference Documents

Project description document: PowerEnJoy Requirement Analysis and Specification

Document (RASD)

Design description Document: Design Document (DD)

Integration Test Plan Document: Testing Document (TD)

Assignments Document: Assignments AA 2016-2017.pdf

2 Project size, cost and effort estimation

This section provides some estimations about the cost, size and necessary effort regarding the development of the PowerEnJoy system. In order to estimate, in a reasonable way, the size of our project we used the Function Points Approach. We have taken in consideration all the main functionalities of our system estimating the correspondent amount of code written in Java. This estimation has been made only on the business logic part, without referring at the various aspects regarding the User Interface. For the estimation of the cost and effort we relied on the COCOMO approach, using as starting point the amount of code we estimated with the Function Points.

2.1 Size estimation: function points

The function points are used to provide an estimation of the project's size. In order to do this estimation each functionality that the system will provide needs to be analyzed by each member of the development team which will provide information regarding its complexity. When the complexity of the functionality has been analyzed it will then be translated in function points. The estimations are based on figures obtained by the statistical analysis of real projects and they are summarized in the following tables.

For Internal Logic Files and External Logic Files

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

For External Output and External Inquiry

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

For External Input

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

UFP Complexity Weights

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

2.1.1 Internal Logic Files (ILs)

PowerEnjoy relies on multiple ILs to store and manage informatios in order to provide the required functionalities. Here we are going to list all the ILs we have identified:

User Information: the system has to store data regarding the registered users, all the information are stored in a single table which contains first name, last name and birthdate together with his account balance, home address, SSN, driving license number, driving license expiring date, history of all the reservations, and phone number and email as contacts. This is a basic file so its complexity has been set as low.

Operator Information: the system also has to store data regarding each operator, all the information are stored in a single table (such as the user information) which contains first name, last name and birthdate together with his SSN, driving license number, driving license expiring date and phone number and email as contacts. This is a basic file so its complexity has been set as low.

Car Information: they are stored using a two-level structure. The first level holds the car status (in use, reserved, available, …), the car plate number, the last revision, the remaining battery, while a secondary table holds the information about the car position as coordinates (<latitude, longitude>) for convenience. This is a basic file so its complexity has been set as low.

Safe Area Information: they are also stored using a two-level structure. The first level holds the type of safe area, the number of available plugs (if there are any), the safe area id, the number plates of the parked cars, while a secondary level holds the information about the area boundaries as coordinates (<latitude, longitude>) necessary to identify the vertices of the zone polygon. This is a basic file so its complexity has been set as low.

Payment Information: this data is stored in a single table. It contains the user SSN, the payment method selected and the credit card/online payment service information. This data is particularly sensitive so, in order to guarantee the system security, needs to be encrypted. This is a particularly complex file to manage, so its complexity has been set to high.

Reservation Information: the reservation data is stored in a single table which holds the identifier of the user who booked them, the ending time, the plate number of the selected car, the safe area in which the car is parked, the car coordinates (<latitude, longitude>). This is a basic file so its complexity has been set as low.

Login Information: the information regarding the login are contained in a single table, that contains both user and operator information to login which are username and password. This is a basic file so its complexity has been set as low.

To summarize all the identified function points we listed them in this table:

Internal Logic Files (ILs)	Complexity	FPs
User Information	Low	7
Operator Information	Low	7
Car Information	Low	7
Safe Area Information	Low	7
Payment Information	Average	15
Reservation Information	Low	7
Login Information	Low	7
Total		57

2.1.2 External Logic Files (ELs)

There are two external data sources PowerEnJoy relies on and they are represented by the Payment Service and the License Verification Service. The interaction between the core system and the payment service provider happens through APIs that encrypts the user data in order to guarantee the security of our system, data can then be returned in a XML format. The results have then to be processed before they can be used in our system. There is one main kind of interaction:

- Given the desired payment method and payment infos, decrypt the files and then recharge the account

For the amount of data that is exchanged in this operation and its complexity , it is reasonable to classify it as a complex one. The interaction between the core system and the license verification service provider is more simple than the previous one. It happens through APIs that only check the validity of the selected driving license, the license data is sent to the service and then it returns a validation string that can be easily processed by the system. For this factors the complexity of this file has been set to low.

External Logic Files (ELs)	Complexity	FPs
Payment Service Data License Verification Service Data	High Low	10 5
Total		15

2.1.3 External Inputs (EIs)

The PowerEnJoy system can be accessed by various types of users and those provide to the system a wide variety of inputs, that we are going to list for each type of user:

All the people:

- Login and Logout: these are basic operations that need an interaction between a little number of components (the application and the user controller, for example). Therefore they have a low complexity and they contribute 3 FPs each.

Users:

- Insert and update their personal information: These are operations of average complexity because they involve sensible information and some of that may need a special treatment in terms of checking the validity and correctness of data. For example the insertion of an updated driving license requires the checking of that license with the Department of Motor Vehicles of the state. In general, though, the updates involve simpler data to be managed. These two operations contribute 4 FPs each.

- Register to the system: This is an average complexity operation because it contains the registration data that needs to be checked, like the operation described above. It then introduces 4 FPs.
- Insert information regarding a location: This is a straightforward operation, a simple insertion of numbers that can happen by hand or with the GPS sensor of the user's device. The low complexity implies that this input counts 3 FPs.
- Request to be notified when a car becomes available in a specified area: This is a complex operation that involves cars' information, safe areas, GPS location and the registration to a service that will send the user a notification. Therefore this operation introduces 6 FPs to the total UFPs.
- Select cars to reserve: The selection of a car is of low complexity, because it is just a simple insertion of data through an abstract representation of it. This contributes 3 FPs.
- Make reservations: This operation is meant to gather all the choices made from the users (car and location) and some hidden information (time, user balance) and check if a reservation can be made. It requires the interaction between an high number of components as it can be seen in the sequence diagrams reported in the design document. This introduces 6 FPs.
- Insert payment information: This is an average operation because it involves the insertion of text and numbers that will be managed by a single component (the payment controller), that will verify if the inserted information are correct. Thus it contributes 4 FPs.
- Unlock cars: This operation is of average complexity because it involves the insertion of the GPS location of the user, and needs to check the status of the car to be unlocked. This is not very complex since the interaction between internal components is really limited and the car to be unlocked can only be the single one reserved by the user. This kind of input operation contributes 4 FPs.

Operators:

- Accept tasks: This is a straightforward operation, the operator just needs to confirm that he has taken care of the task submitted by the dispatcher. It introduces 3 FPs.

Dispatcher:

- Selects operators: This is a low complexity input operation, because it is a simple selection of data in a list of operators. It contributes 3 FPs.
- Selects cars: As the selection of the operators this is not a complex operation. 3 FPs are introduced.
- Creates tasks: This operation is more complex than the other two, because it requires the involvement of different components and objects (cars, operators) and will change their status, so this operation needs to be controlled for the correct functioning of the system. 6 FPs are introduced by this input operation.

The system also constantly receives information from the sensors of the cars, but this is a simple “receive and store” routine, but it involves a massive amount of transferred data, hence the complexity of the operation cannot be just low. We consider that the database is built in a way that optimizes the effort spent to transfer all that data, so that this operation becomes of average complexity and contributes 4 FPs.

To summarize all the identified function points we listed them in this table:

External Inputs (EI)	Complexity	FPs
Login and Logout	Low	3x2
Insert and update of personal information	Average	4x2
Registration to the application	Average	4
Insert information regarding a location	Low	3
Request to be notified	High	6
Select a car to reserve	Low	3
Make a reservation	High	6
Insert payment information	Average	4
Unlock car	Average	4
Accept task	Low	3
Select operator	Low	3
Select car	Low	3
Create task	High	6
Sensor data from cars	Average	4
Total		63

2.1.4 External Inquiries (EQs)

Users can request data to the PowerEnJoy system in order to know what is their current status in the system and what they are supposed to do. These are the main requests users can make:

Registered users:

- Request their reservation history: This kind of interaction between the application and the user is quite simple, because the system needs to retrieve a single type of information, and the user can request it in few simple passages. The low complexity of this operation implies the introduction of 3 FPs. The same kind of reasoning can be applied to the other inquiries made by the user.
- Request their personal information stored on the application.
- Request their balance on PowerEnJoy.
- Request information regarding their current reservation.

- Request the list of cars in a specific area, with their position and status: This interaction with the system is more complex because it requires the application to compute a list of cars, with their information, based on their status and location. With the help of the right algorithm this operation can be not too costly in terms of computation. This introduces 4 FPs.

Operators:

- Request the of their current tasks.
- Request their active task's information.

The two inquiries are just related to information retrieval, so they are simple to carry out and contribute 3 FPs to the total count.

Dispatcher:

- Request the working operators' information
- Request all the cars' information

The two inquiries are just related to information retrieval, so they are simple to carry out and contribute 3 FPs to the total count.

To summarize all the identified function points we listed them in this table:

External Inquiries (EQ)	Complexity	FPs
Request their reservation history	Low	3
Request their personal information	Low	3
Request their balance on PowerEnJoy	Low	3
Request their current reservation	Low	3
Request the list of cars in a specific area, with their position and status	Average	4
Request the list of their current tasks	Low	3
Request their active task's information	Low	3
Request the working operators' information	Low	3
Request all the cars' information	Low	3
Total		26

2.1.5 External Outputs (EOs)

The PowerEnJoy system is in charge of notifying users and operators when some changes in objects of the system happen. The cases are:

- Notification of a new available car to users who submitted for this kind of information: This is the only non trivial interaction between the system and the users that can be defined as output, because it involves the change in a car's status, the verification that the car can be used, the system needs to notify the user or the group of users and waits for an answer. In some cases the users will then reserve that car. So this operation is of average complexity and implies the introduction of 5 FPs.
- Notification of a lost reservation to defaulting users.
- Notification of an accepted reservation to users.
- Notification of a refused reservation to users.
- Notification of a successful change in the user's information.
- Notification of an accepted payment to users.
- Notification of a new task to operators.

Except the first one in the previous list, all the external outputs are quite simple operations and contribute 4 FPs each.

To summarize all the identified function points we listed them in this table:

External Outputs (EO)	Complexity	FPs
Notification of a new available car Notification of a lost reservation Notification of an accepted reservation Notification of a refused reservation Notification of a successful change in the user's information Notification of an accepted payment Notification of a new task	Average Low Low Low Low Low Low Low	5 4 4 4 4 4 4 4
Total		37

2.1.6 Overall Estimation

Function Type	Value
Internal Logic Files External Logic Files External Inputs External Inquiries External Outputs	57 15 63 26 37
Total	198

Following the estimate of the average Lines Of Code per Function Point for Java applications we have found in the book “Applied Software Measurement, Caper Jones, McGraw Hill, 1996”, that is 53 LOC/FP, we estimated the number of Source Lines Of Code of the application:

$$SLOC = 53 \times 198 = 10494$$

2.2 Cost and effort estimation: COCOMO II

To estimate the cost and the effort of the development of the PowerEnJoy application we are going to use the COCOMO II model.

2.2.1 Scale Factors

In the following table there is the list of the 5 scale factors, that we are going to analyze, and their weights that will be used in the effort formula, together with a brief description.

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC SF1	thoroughly unprece-dented 6.20	largely unprece-dented 4.96	somewhat unprece-dented 3.72	generally familiar 2.48	largely familiar 1.24	thoroughly familiar 0.00
FLEX SF2	rigorous 5.07	occasional relaxation 4.05	some relaxation 3.04	general conformity 2.03	some conformity 1.01	general goals 0.00
RESL SF3	little (20%) 7.07	some (40%) 5.65	often (60%) 4.24	generally (75%) 2.83	mostly (90%) 1.41	full (100%) 0.0

TEAM SF4	very difficult interac-tions 5.48	some difficult interac-tions 4.38	basically cooperati-ve interac-tions 3.29	largely cooperati-ve 2.19	highly cooperati-ve 1.10	seamless interac-tions 0.00
PMAT SF5	SW-CMM Level 1 Lower 7.80	SW-CMM Level 1 Upper 6.24	SW-CMM Level 2 4.68	SW-CMM Level 3 3.12	SW-CMM Level 4 1.56	SW-CMM Level 5 0.00

- Precedentness: Since we don't have much experience in developing projects of this kind we chose the value "Low".
- Development flexibility: We don't have any freedom to deviate from the requirements and create something different, but in the assignments of the projects some things are not specified. For example the architecture, the tools that we should use, the devices that we should grant support to. Anyway we have to observe the requirements, so, we chose the value "Low" for this scale factor.
- Risk resolution: Looking at our risk analysis (written in this document) and being a little pessimistic, we decided to choose the value "Very High" for this scale factor.
- Team cohesion: We are a very collaborating team, we worked together in other projects in the past with good results. Thus we decided to choose the value "Very High".
- Process maturity: Since this is the first time we deal with a project of this complexity and we are still learning as the lectures go on, we feel like the best fitting in the CMM hierarchy is "Level 2".

To aggregate the 5 scale factors we sum up our reasoning in this table:

Scale Factor	Factor	Value
Precedentness (PREC) Development flexibility (FLEX) Risk resolution (RESL) Team cohesion (TEAM) Process maturity (PMAT)	Low Low Very High Very High Level 2	4.96 4.05 1.41 1.10 4.68
Total		16.2

2.2.2 Cost Drivers

- **Required Software Reliability:**
  If the system goes down for some days the image of the company could result ruined, and the company could lose its customers (as well as the loss income for those days). It is very important that the system runs without problems, at least during the daytime. So, in our case the cost driver is set to "High".

RELY Cost Driver						
RELY Descrip-tors	slightly inconve-nience	easily recovera-ble losses	moderate recovera-ble losses	high financial loss	risk to 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:**
  We don't have realistic estimates of the database size, but from our experience and keeping in mind that our system needs to manage data coming from an high number of cars, and that PowerEnJoy might expand its boundaries and reach many cities, we feel safe to assume that this cost driver can be set to "High".

DATA Cost Driver						
DATA Descrip-tors		$\frac{D}{P} < 10$	$10 \leq \frac{D}{P}$ $\frac{D}{P} \leq 100$	$100 < \frac{D}{P}$ $\frac{D}{P} \leq 1000$	$\frac{D}{P} > 1000$	
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:**
  This cost driver is set to "Very High" because the PowerEnJoy system needs to interact with an heterogeneous set of objects and users, with different kinds of interactions. So it has to manage complex kinds of information. In the future this product might be expanded and updated many times to include other functionalities and compatibility with other devices, objects and physical locations.

CPLX Cost Driver						
Rating level	Very Low	Low	Nominal	High	Very High	Extra High
Effort multipliers	0.73	0.87	1.00	1.17	1.34	1.74

- **Required Reusability**
  Since the requirements do not state anything about the reusability of the product, we set this value to “Nominal”.

RUSE Cost Driver						
RUSE Descrip-tors		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:**
  Everything regarding the life-cycle of the project is written in the documents, so we set this cost driver to “Nominal”.

DOCU Cost Driver						
DOCU Descrip-tors	Many life-cycle needs uncovered	Some life-cycle needs uncovered	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very ex-cessive 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:**
  From the previously described complex interactions that the system should support, we think that we can set this cost driver to "High".

TIME Cost Driver						
TIME Descrip-tors			≤50% use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating 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:**
  To maintain the performances of the system we need the storage to be fast and big enough, so our system will have this cost driver set to "Nominal".

STOR Cost Driver						
STOR Descrip-tors			≤50% 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:**
  Like other modern products that involve the use of mobile applications we will distribute minor updates every 2 weeks and major updates every 6 months. Bug fixes may happen more frequently. So our choice for this cost driver is the value "Nominal".

PVOL Cost Driver						
PVOL Descrip-tors		Major change every 12 mo., minor change every mo.	Major: 6 mo., minor: 2 wk.	Major: 2 mo, minor: 1 wk	Major: 2 wk, 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 are still students and, even though our proposed solution might not be perfect and might not describe the system fully, we think that our analysis of the project has been pretty decent. Therefore this cost driver is set to “High”.

ACAP Cost Driver						
ACAP De-scriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th 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**
  Looking at our programming skills and given that we are not expert in the JEE framework we set this cost driver to “Nominal”.

PCAP Cost Driver						
PCAP De-scriptors	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 some experience in Java programming but not much in JEE programming, so we set this cost driver to "Nominal".

APEX Cost Driver						
APEX De-scriptors	≤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 much experience with JEE, but we have already worked on some Java projects and we have some familiarity with DMBSs, so we set this cost driver to "Nominal".

PLEX Cost Driver						
PLEX De-scriptors	≤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:**
  As for the previous cost driver we set this one to "Nominal".

LTEX Cost Driver						
LTEX De-scriptors	≤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:**
  We don't have much time to work on and complete this project, also we have a time limit of a semester so we set this cost driver to the value "Very Low".

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:**
  We have a complete framework of tools that work nicely together and cover all the functionalities that we need to develop and manage the system's life-cycle. So we set this cost driver's value to "High".

TOOL Cost Driver						
TOOL Descriptors	edit, code, debug	simple, frontend, backend CASE, little integration	basic life-cycle tools, moderately integrated	strong, mature life-cycle tools, moderately integrated	strong, mature, proactive life-cycle tools, well integrated with processes, methods, reuse	
Rating 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 have been working on this project in parallel and we needed to be well coordinated, so we had meetings, we had a lot of conversations on VOIP programs and also desktop-sharing conferences. So we set this cost driver to "Very High", even though we live in different towns, because it was like being in the same office.

SITE Cost Driver						
SITE Colloca-tion De-scriptors	Interna-tional	Multi-city and multi-company	Multi-city or multi-company	Same city or metro area	Same building or complex	Fully collocated
SITE Com-munica-tions De-scriptors	Some phone, mail	Individual phone, fax	Narrow band email	Wideband electronic communi-cation	Wideband elect. comm., occasional video conf.	Interactive multime-dia
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:**
  This cost driver is set to “High” because the documentation writing part of the project has requested a lot of effort, in scheduling the meetings and the working hours, as well as delivering the project’s documents.

SCED Cost Driver						
SCED De-scriptors	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 summarizes our choices, making it clear which are all the effort multipliers to substitute in the effort formula:

Cost Driver	Factor	Value
Required Software Reliability (RELY) Database Size (DATA) Product Complexity (CPLX) Required Reusability (RUSE) Documentation match to life-cycle needs (DOCU)	High High Very High Nominal Nominal	1.10 1.14 1.34 1 1

Execution Time Constraint (TIME)	High	1.11
Main Storage Constraint (STOR)	Nominal	1
Platform Volatility (PVOL)	Nominal	1
Analyst Capability (ACAP)	High	0.85
Programmer Capability (PCAP)	Nominal	1
Application Experience (APEX)	Nominal	1
Platform Experience (PLEX)	Nominal	1
Language and Tool Experience (LTEX)	Nominal	1
Personnel Continuity (PCON)	Very Low	1.29
Usage of Software Tools (TOOL)	High	0.90
Multisite Development (SITE)	Very High	0.86
Required Development Schedule (SCED)	High	1
Total		1.582978

2.2.3 Effort equation

The estimation of the effort, in Person-Months, is given by the following equation:

$$Effort = A \times Size^{E} \times \prod_{i=1}^{17} EMi$$

The terms in this equation are the following:

$A = 2.94$ for COCOMO II.2000

$$E = B + 0.01 \times \sum_{J=1}^{5} SFj$$

$\square = 0.91$ for COCOMO II.2000

$$E = 0.91 + 0.01 \times 16.2 = 1.072$$

$$Size = 10.494\ KSLOC$$

So, the final result is:

$$Effort = 2.94 \times 10.494^{1.072} \times 1.582978 = 57.85\ PM \approx 58\ PM$$

2.2.4 Schedule estimation

The COCOMO II schedule equation predicts the number of months required to complete our software project. The duration is based on the effort predicted by the effort equation:

$$Duration\ = 3.67 \times Effort^{SE}$$

Where:

Effort is the effort from the COCOMO II effort equation:

$$Effort = 57.85\ PM$$

SE is the schedule equation exponent derived from the five Scale Drivers:

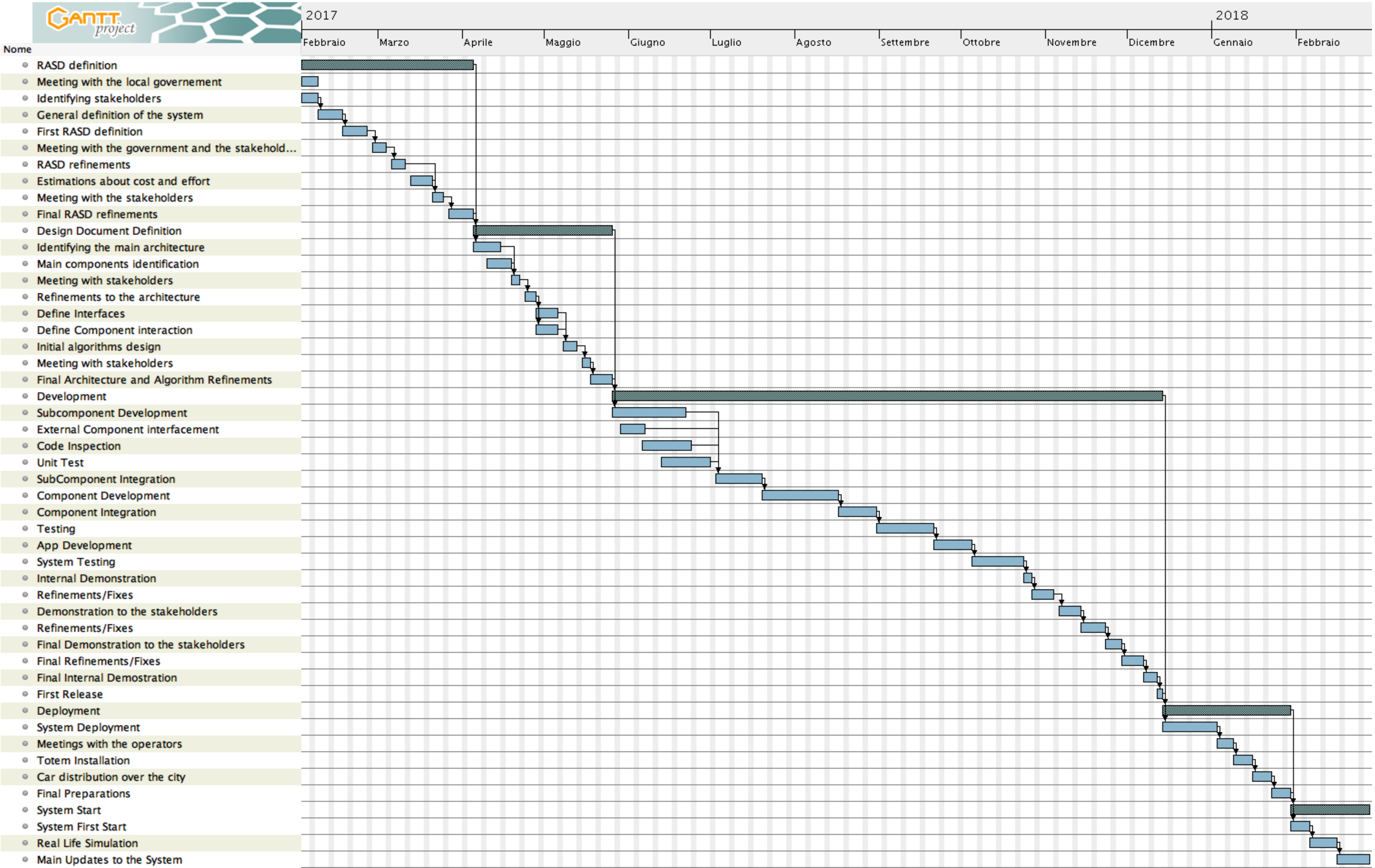
$$SE = 0.28 + 0.2\times(E - B) = 0.28 + 0.2\times0.162 = 0.3124$$

Hence, the estimated duration is:

$$Duration = 3.67\times57.85^{0.3124} = 13{,}04\ Months$$

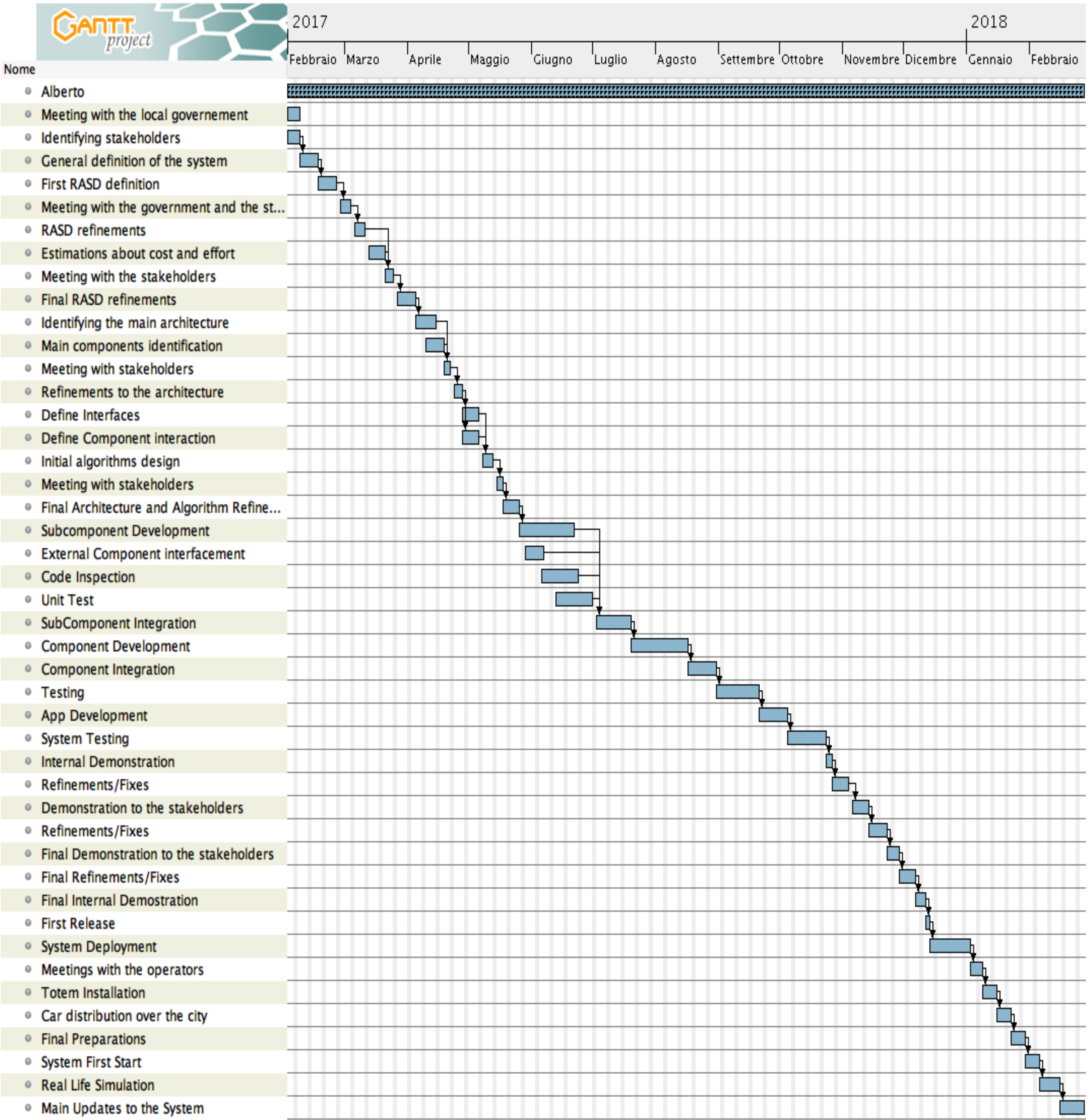
3 Schedule

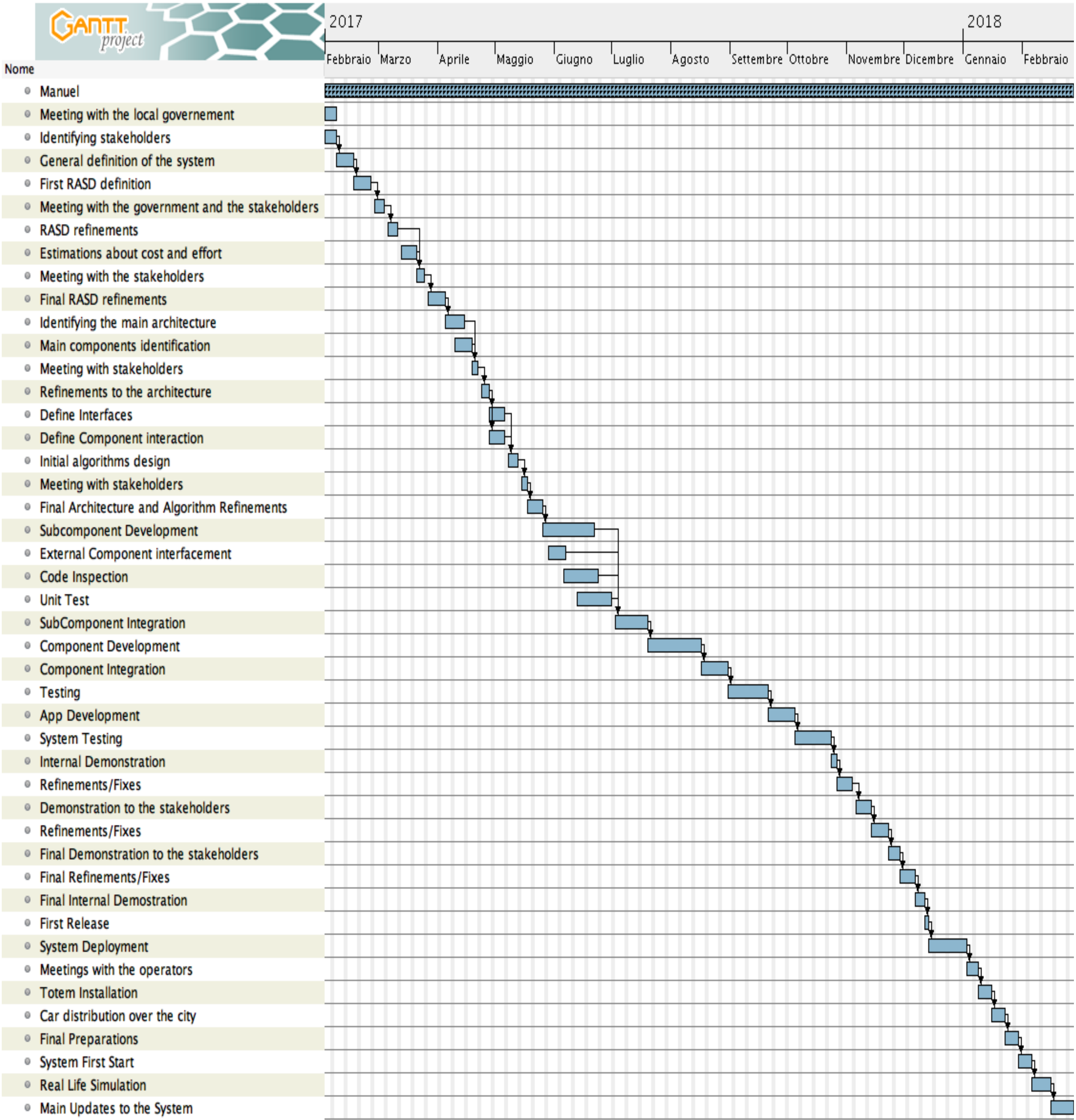
Here we are going to provide a general approximation of the project schedule. This includes the various stages of the system developing since the first embryonic stages, such as the early system definitions, to the last stages of development and deployment. It’s important to remember that this is a general overview of the schedule, therefore there we’ll be refinements in the schedule, and a more specific task planning, as the project progresses in development.


2017
2018
Febbraio
Marzo
Aprile
Maggio
Giugno
Luglio
Agosto
Settembre
Ottobre
Novembre
Dicembre
Gennaio
Febbraio
Nome
RASD definition
Meeting with the local governement
Identifying stakeholders
General definition of the system
First RASD definition
Meeting with the government and the stakehold...
RASD refinements
Estimations about cost and effort
Meeting with the stakeholders
Final RASD refinements
Design Document Definition
Identifying the main architecture
Main components identification
Meeting with stakeholders
Refinements to the architecture
Define Interfaces
Define Component interaction
Initial algorithms design
Meeting with stakeholders
Final Architecture and Algorithm Refinements
Development
Subcomponent Development
External Component interfacement
Code Inspection
Unit Test
SubComponent Integration
Component Development
Component Integration
Testing
App Development
System Testing
Internal Demonstration
Refinements/Fixes
Demonstration to the stakeholders
Refinements/Fixes
Final Demonstration to the stakeholders
Final Refinements/Fixes
Final Internal Demostration
First Release
Deployment
System Deployment
Meetings with the operators
Totem Installation
Car distribution over the city
Final Preparations
System Start
System First Start
Real Life Simulation
Main Updates to the System

4 Resource Allocation

In this section we are going to provide an explanation on how the tasks described above are associated to each team member. For the sake of readability the diagrams are two, one for team member:




GANTT project
Nome
2017
2018
Febbraio
Marzo
Aprile
Maggio
Giugno
Luglio
Agosto
Settembre
Ottobre
Novembre
Dicembre
Gennaio
Febbraio
Manuel
Meeting with the local governement
Identifying stakeholders
General definition of the system
First RASD definition
Meeting with the government and the stakeholders
RASD refinements
Estimations about cost and effort
Meeting with the stakeholders
Final RASD refinements
Identifying the main architecture
Main components identification
Meeting with stakeholders
Refinements to the architecture
Define Interfaces
Define Component interaction
Initial algorithms design
Meeting with stakeholders
Final Architecture and Algorithm Refinements
Subcomponent Development
External Component interfacement
Code Inspection
Unit Test
SubComponent Integration
Component Development
Component Integration
Testing
App Development
System Testing
Internal Demonstration
Refinements/Fixes
Demonstration to the stakeholders
Refinements/Fixes
Final Demonstration to the stakeholders
Final Refinements/Fixes
Final Internal Demostration
First Release
System Deployment
Meetings with the operators
Totem Installation
Car distribution over the city
Final Preparations
System First Start
Real Life Simulation
Main Updates to the System

5 Risk Management

In this section of the document we are going to focus on the main risks we might encounter during the project development. Some of them could be caused by technical issues, while others could be related by financial or management problems. For the sake of readability and comprehension all the risks are listed in this table, each risk has a description, the probability we think it might occur (low, moderate, high, critical), the impact that it will do if it does occur (low, moderate, high, critical) and a simple contingency plan.

Risk	Probability	Impact	Contingency plan
A member of the development team cannot continue working on the project	Low	Critical	Reschedule the project planning to minimize the delay caused by the absent member
Malfunctioning of one of the development platforms used for our project	Moderate	Moderate	Buy or rent another equivalent, and possibly more reliable, platform that allow us to complete all the tasks
Some of the stakeholders cry off and decide to not finance the project anymore	Moderate	Critical	Try to find other stakeholders interested in the project, or, in case we don't find any, resize the project in order to minimize the less caused by the absent funds
JEE platform malfunctions and the problem prevent further development of the project	Moderate	High	Obtain assistance from some technician and find a different way to maintain the code while the service is being reestablished

Something regarding the installment of the service in the selected city goes wrong	Moderate	High	Change, adapt the project in order to meet the government specifications and minimize the loss in terms of functionalities

6 Additional Information

6.1 Hours of Work

Assignment	Hours per person
Requirement Analysis And Specification Document (RASD)	30
Design Document (DD)	22
Integration Testing Plan Document (ITPD)	20
Project Planning Document (PPD)	25
Code Inspection Document	7
Total	104

6.2 Used Tools

The tools that supported us are:

MS Office Word 2016: to write and redact this document;

GanttProject: to produce the gantt diagrams;