POLITECNICO
MILANO 1863

Politecnico di Milano

Software Engineering 2 project

PowerEnJoy

Project Plan Document

Matteo Penco
mat. 875740

Riccardo Pressiani
mat. 874948

Version 1.1 approved

February 7, 2017

Revision History

Revision	Date	Author(s)	Description
0.1	18.01.17	RP	Document created
1.0	22.01.17	RP and MP	Document approved
1.1	07.02.17	MP	Fix Gantt diagrams

Hours of work

Matteo Penco	15h
Riccardo Pressiani	15h

Contents

Chapter 1

Introduction

1.1 Purpose

The Project Plan Document provides an high detailed description of the PowerEnJoy project planning. First of all, an estimation of the effort related to the project development will be provided. Then, a description of the tasks and the resource allocation will be given. Finally a risk analysis will be conduct.

The intended readers of this document are all the project stakeholders, that also include the QA and development team. It is important that the information contained in this document are fully understood in order to deliver to product on time. Moreover, this document can be useful even for the local governments and all those entities that could be interested to sponsor the project.

1.2 Scope

PowerEnJoy is inteded to be a management system for a car-sharing system that exclusively employs electrical powered cars. The system allows users to find all the available cars near a given location which can be their current position or a specific address typed in. The user can book one of the cars available for a limited amount of time. Once the booked car is reached by the user, it can be unlocked from one of the smart devices of the user. After entering a PIN, decided by the user during the registration phase, on the onboard computer, the engine can be started and the rental begins. The cost of the service is calculated on the total duration of the rental multiplied by a fixed rate per minute. The user is continuously informed about the cost of the ongoing rental by the onboard computer. The user can end the rental by parking the car and stopping the engine.

The PowerEnJoy platform is intended to be available on the major mobile devices, such as smartphones and tablets running iOS [1] or Android [2].

1.3 Definitions, acronyms and abbreviations

1.3.1 Definitions

Onboard computer An onboard computer is intended to be the embedded device integrated in the car system that provide two main functions: on one hand it shows the user basic information about the ongoing rental, on the other hand it is in charge of all the communications related to the state of the car between the car and the PowerEnJoy management system.

Safe area A safe area is an area, within prede

ned edges, in which the users are allowed to park the rented cars. The users is not allowed to park, and therefore end a rental, if he/she is not inside a safe area.

Component Basing on the definition given in UML 2, a component is considered an autonomous, encapsulated unit within a system or subsystem that provides one or more interfaces. The component diagram of the system is provided in the Design Document.

1.3.2 Acronyms

AVC Average Variable Cost

COCOMO COnstructive COst MOdel

DD Design Document

EI External Input

EM Effort Multiplier

ELF External Logic File

EO External Output

EQ External Inquiry

FP Function Point

ILF Internal Logic File

KSLOC Kilo-Source Lines Of Code

PM Person-Months

QA Quality Assurance

RASD Requirements Analysis and Specification Document

SLOC Source Lines Of Code

SW-CMM Software Capability Maturity Model

UFP Unadjusted Function Point

1.3.3 Abbreviations

PREC Precedentedness

FLEX Development Flexibility

RESL Architecture / Risk Resolution

TEAM Team Cohesion

PMAT Process Maturity

RELY Required Software Reliability

DATA Database Size

CPLX Product Complexity

RUSE Required Reusability

DOCU Documentation match to life-cycle needs

TIME Execution Time Constraint

STOR Storage Constraint

PVOL Platform Volatility

ACAP Analyst Capability

PCAP Programmer Capability

APEX Application Experience

PLEX Platform Experience

LTEX Language and Tool Experience

PCON Personnel Continuity

TOOL Usage of Software Tools

SITE Multisite Development

SCED Required Development Schedule

24/7 24 hours per day, 7 days per week.

1.4 References

- Assigments document for the first semester project of the Software Engineering 2 course held at Politecnico di Milano by Mottola Luca and Di Nitto Elisabetta [3].
- PowerEnjoy - Requirement Analysis and Specification Document [4].
- PowerEnjoy - Design Document [5].
- PowerEnjoy - Integration Test Plan Document [6].

1.5 Overview

In this Chapter a first introduction of this Project Plan Document has been provided. In the following Chapters several aspects related to the PowerEnJoy project planning will be discussed.

In Chapter 2, an estimation of the expected size, cost and required effort of the project development will be provided. The Function Points approach will be exploited for the size estimation, while COCOMO II will be the approach used for the cost and effort estimation.

In Chapter 3, we are going to provide, thanks to a Gantt chart, a high-level project schedule, defining all main tasks necessary for developing the project.

In Chapter 4, we are going to provide, thanks to a Gantt chart, a general overview of how resources (in this case the two members of the team) have been assigned to various tasks of the project.

In Chapter 5, an analysis of the risk prevention and management will be provided. The risks identified and related to planning, technical and business issues and the possible strategies to overcome them will be discussed

Chapter 2

Project size, cost and effort estimation

This chapter provides an estimation of the expected size, cost and required effort of the PowerEnJoy project.

The size estimation will be based on the Function Points approach. This approach estimates the size of a software project starting from the functionalities that it has to offer. It is important to state that only the business logic has been taken into account, while the user applications have not been included in the estimation.

The cost and effort estimation will be based on the COCOMO II approach, starting from the SLOC value obtained from the size estimation.

2.1 Size estimation: Function Points

The Function Points approach estimates the size of a software project starting from the functionalities that it has to offer. Several real projects have been analyzed to determine the number of FPs associated with a set of Function Types and the related estimated complexity. The values are provided by the table below:

Table 2.1: 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 Inquiries	3	4	6
External Outputs	4	5	7

2.1.1 Internal Logic Files

ILFs are groups of homogeneous data or control information that are stored within the application. In the following paragraphs, a brief discussion about the PowerEnJoy ILFs will be provided.

First of all, the users data need to be stored in the system. A User entity in the database stores personal details (name, surname and tax code), the driver licence identification number, login data and payment information. All these items are stored as strings.

For what concerns the Car status entities, the items that need to be stored are the identification number, the status of the car (Available, Unavailable, Reserved and In Use) and the position as a structure that includes the latitude and longitude values.

The same structure used to store the position of the cars is used in the Safe Area entities. This information allows the system to compare the position of a car, when a rent has just ended, with the ones that represent the safe areas. Moreover, a boolean value indicates the possibility or not to plug the cars with the power grid in a specific safe area.

The last entities that need to be stored are the ones related to the bookings and rentals. The former stores the starting time of the booking and if that booking has expired or not. Moreover, it has a reference to the user that started the booking and the car that has been booked. The latter, instead, stores the rental duration and, at the end of the rental, the amount due by the driver is included.

The previous considerations and the UFP Complexity Weights provided in the Table 2.1 has led to the following ILFs Function Points estimation:

Table 2.2: ILFs Function Points

ILF	Complexity	FPs
User data	Low	7
Car status	Low	7
Safe areas	Low	7
Booking	Average	10
Rental	Average	10
Total		41

2.1.2 External Logic Files

ELFs are groups of homogeneous data used by the application to be developed but retrieved and created by external applications.

PowerEnJoy needs to exchange messages with the following services:

- **Recovery service** The Recovery Service is required in order to advise an external company when a car requires attention because a non expected event has happened.

- **Notification service** The Notification Service is required in order to notify the user via the PowerEnJoy app or the onboard computer integrated in the cars of events not triggered directly by the user (e.g. the expiration of a booking).

- **Payment service** The Payment Service is required in order to charge the user with the due amount related to the PowerEnJoy services.

The communications between PowerEnJoy and external application are executed through RESTful APIs and since simple strings needs to be transmitted it is expected that the data structure will be in JSON format.

The previous considerations and the UFP Complexity Weights provided in the Table 2.1 has led to the following ELFs Function Points estimation:

Table 2.3: ELFs Function Points

ELF	Complexity	FPs
Recovery request	Low	5
Notification message	Low	5
Payment request	Low	5
Total		15

2.1.3 External Inputs

EIs are all the operations offered by the application that elaborate data coming from the external environment. These operations can be triggered by the users from both the PowerEnJoy application and the onboard computer.

Based on the estimated complexity of the EIs and the UFP Complexity Weights provided in the Table 2.1, the following table provides the Function Points estimation:

Table 2.4: EIs Function Points

EI	Complexity	FPs
User registration	Average	4
Login/Logout	Low	2*3
Book a car	Average	4
Delete booking	Low	3
Unlock car doors	Average	4
Begin rental	High	6
End rental	High	6
Total		33

If the reader wants to examine in depth the functionalities mentioned in the table above, more details can be found in the RASD [4] and DD [5].

2.1.4 External Inquiries

EQs are all the operations offered by the application that involves input and ouput data. This means that this kind of operations are are triggered by the users and a response with the data required is provided through the PowerEnJoy application or the onboard computers.

Based on the estimated complexity of the EQs and the UFP Complexity Weights provided in the Table 2.1, the following table provides the Function Points estimation:

Table 2.5: EQs Function Points

EQ	Complexity	FPs
Find cars	Low	3
Get details about car	Low	3
Total		6

If the reader wants to examine in depth the functionalities mentioned in the table above, more details can be found in the RASD [4] and DD [5].

2.1.5 External Outputs

EOs are all the operations offered by the application that generates data for the external environments. These operations are triggered by the PowerEnJoy backend system and sended to the users via the PowerEnJoy application and or the onboard computers.

Based on the estimated complexity of the EOs and the UFP Complexity Weights provided in the Table 2.1, the following table provides the Function Points estimation:

Table 2.6: EOs Function Points

EO	Complexity	FPs
Elapsed booking notification	Low	4
Unlock car doors notification	Low	4
Ended rental notification	Low	4
Get info about ongoing rental	High	7
Total		19

If the reader wants to examine in depth the functionalities mentioned in the table above, more details can be found in the RASD [4] and DD [5].

2.1.6 Overall Estimation

The following table provides the overall estimation of all the Function Types discussed above:

Table 2.7: Function Points overall estimation

Function Type	Value
Internal Logic Files	41
External Logic Files	15
External Inputs	33
External Inquiries	6
External Outputs	19
Total	114

Below, the expected SLOC, calculated using the formulas provided by the Function Points method, is provided. The value has been calculated considering Java as the development language for the business logic part of the PowerEnJoy system (see Section 2.4 of DD [5] for more details).

$$SLOC = AVC * number\ of\ function\ points$$

Given the Java AVC values provided by QSM [7], the estimated SLOC lower bound is:

$$SLOC = 14 * 114 = 1596$$

While the upper bound is estimated to be:

$$SLOC = 134 * 114 = 15276$$

In the following section the SLOC value that will be considered is the one calculated using the AVC average value:

$$SLOC = 53 * 114 = 6042$$

2.2 Cost and effort estimation: COCOMO II

In this section we're going to use COCOMO II approach to estimate the cost and effort needed for developing PowerEnJoy system.

2.2.1 Scale Drivers

For estimating scale drivers values we refer to the official COCOMO II table:

Table 2.8: Scale Drivers values for COCOMO II Model

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	**largely unprecedented**	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
SF_j	6.20	**4.96**	3.72	2.48	1.24	0.00
FLEX	rigorous	**occasional relaxation**	some relaxation	general conformity	some conformity	general goals
SF_j	5.07	**4.05**	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	**mostly (90%)**	full (100%)
SF_j	7.07	5.65	4.24	2.83	**1.41**	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	**highly cooperative**	seamless interactions
SF_j	5.48	4.38	3.29	2.19	**1.10**	0.00
PMAT	SW-CMM Level 1 Lower	SW-CMM Level 1 Upper	SW-CMM Level 2	**SW-CMM Level 3**	SW-CMM Level 4	SW-CMM Level 5
SF_j	7.80	6.24	4.68	**3.12**	1.56	0.00

Here a brief description of scale drivers considered above:

- Precedentedness - It reflects the previous experience of the team with the development of similar projects.

Table 2.9: PREC Rating Levels

Feature	Very Low	Nominal / High	Extra High
Organizational understanding of product objectives	General	**Considerable**	Thorough
Experience in working with related software systems	**Moderate**	Considerable	Extensive
Concurrent development of associated new hardware and operational procedures	**Extensive**	Moderate	Some
Need for innovative data processing architectures, algorithms	Considerable	**Some**	Minimal

- Development Flexibility - It reflects the degree of flexibility in the development process with respect to specific constraints, requirements and external specifications.

Table 2.10: FLEX Rating Levels

Feature	Very Low	Nominal / High	Extra High
Need for software conformance with pre-established requirements	**Full**	Considerable	Basic
Need for software conformance with external interface specifications	Full	**Considerable**	Basic
Combination of inflexibilities above with premium on early completion	High	**Medium**	Low

- Risk Resolution - It reflects the level of analysis with respect to the risk management plan and the definition of budget and schedule.

Table 2.11: RESL Rating Levels

Characteristic	Very Low	Low	Nominal	High	Very High	Extra High
Risk Management Plan identifies all critical risk items, establishes milestones for resolving them by PDR or LCA.	None	Little	Some	Generally	**Mostly**	Fully
Schedule, budget, and internal milestones through PDR or LCA compatible with Risk Management Plan.	None	Little	Some	**Generally**	Mostly	Fully
Percent of development schedule devoted to establishing architecture, given general product objectives.	5	10	17	**25**	33	40
Percent of required top software architects available to project.	20	40	60	80	**100**	120
Tool support available for resolving risk items, developing and verifying architectural specs.	None	Little	Some	Good	**Strong**	Full
Level of uncertainty in key architecture drivers: mission, user interface, COTS, hardware, technology, performance.	Extreme	Significant	Considerable	Some	**Little**	Very Little
Number and criticality of risk items.	> 10 Critical	5-10 Critical	**2-4 Critical**	1 Critical	> 5 Non Critical	< 5 Non Critical

- Team Cohesion - It reflects the level of coordination and synchronization of project's stakholders: users, customers, developers, maintainers, interfacers, others.

Table 2.12: TEAM Rating Levels

Characteristic	Very Low	Low	Nominal	High	Very High	Extra High
Consistency of stakeholder objectives and cultures	Little	Some	Basic	Considerable	Strong	**Full**
Ability, willingness of stakeholders to accommodate other stakeholders? objectives	Little	Some	Basic	Considerable	**Strong**	Full
Experience of stakeholders in operating as a team	None	Little	Little	Basic	Considerable	**Extensive**
Stakeholder team-building to achieve shared vision and commitments	None	Little	Little	Basic	Considerable	**Extensive**

- Process Maturity - It reflects the level of maturity of the software organization.

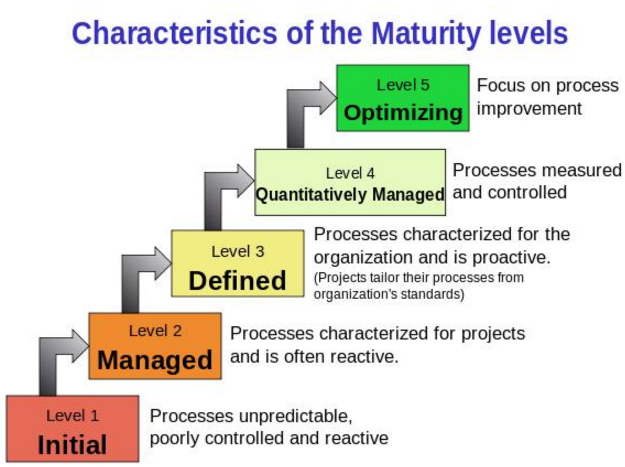


Figure 2.1: The picture above shows levels of process maturity defined by SW-CMM.

Here the result of scale drivers' evaluation:

Table 2.13: Scale Drivers overall estimation

Scale Driver	Factor	Value
Precedentedness (PREC)	Low	4.96
Development flexibility (FLEX)	Low	4.05
Risk resolution (RESL)	Very high	1.41
Team cohesion (TEAM)	Very high	1.10
Process maturity (PMAT)	Level 3	3.12
Total		14.64

2.2.2 Cost Drivers

Required Software Reliability

This is the measure of the extent to which the software must perform its intended function over a period of time. Supposing that electric car sharing will be largely used in the near future and the system will have great success, a potential malfunctioning could lead to high financial losses.

Table 2.14: RELY values

RELY Cost Drivers						
RELY Descriptors	slightly inconvinience	easily recoverable losses	moderate recoverabile 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

This cost driver attempts to capture the effect large test data requirements have on product development. The rating is determined by calculating D/P, the ratio of bytes in the testing database to SLOC in the program. The reason the size of the database is important to consider is because of the effort required to generate the test data that will be used to exercise the program. In the case of PowerEnJoy a 5 Mb testing database has been estimated and for this reason:

$$\frac{D}{P} = 5 * 10^6/6042 = 828$$

Table 2.15: DATA values

DATA Cost Drivers						
DATA Descriptors		$\frac{D}{P} < 10$	$10 < \frac{D}{P} < 100$	$100 < \frac{D}{P} < 1000$	$\frac{D}{P} > 1000$	
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

According to COCOMO II rating scale, the complexity of development of the system is considered high.

Table 2.16: CPLX values

CPLX Cost Drivers						
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

This cost driver accounts for the additional effort needed to construct components intended for reuse on current or future projects. “Across program” could apply to reuse across multiple financial applications projects for a single organization: for example some system’s components could be reused for other ”sharing-systems”.

Table 2.17: RUSE values

RUSE Cost Drivers						
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

The rating scale for the DOCU cost driver is evaluated in terms of the suitability of the project's documentation to its life-cycle needs.

Table 2.18: DOCU values

DOCU Cost Drivers						
DOCU Descriptors	Many life-cycle needs uncovered	Some life-cycle needs uncovered	Right-sized to life-cycle 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

This is a measure of the execution time constraint imposed upon a software system. The rating is expressed in terms of the percentage of available execution time expected to be used by the system or subsystem consuming the execution time resource.

Table 2.19: TIME values

TIME Cost Drivers						
TIME Descriptors			$\leq$ 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

This rating represents the degree of main storage constraint imposed on a software system or subsystem.

Table 2.20: STOR values

STOR Cost Drivers						
STOR Descriptors			$\leq$ 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

"Platform" is used here to mean the complex of hardware and software the software product calls on to perform its tasks; the platform includes any compilers or assemblers supporting the development of the software system. We estimate that a major release of the software will be needed about every 12 months.

Table 2.21: PVOL values

PVOL Cost Drivers						
PVOL Descriptors		Major change every 12 mo., minor change every 1 mo.	Major: 6mo; minor: 2wk.	Major: 2mo, minor: 1wk	Major: 2wk; 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

Analysts are personnel who work on requirements, high-level design and detailed design. The major attributes that should be considered in this rating are analysis and design ability, efficiency and thoroughness, and the ability to communicate and cooperate.

Table 2.22: ACAP values

ACAP Cost Drivers						
ACAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating level	Very Low	Low	Nominal	**High**	Very High	Extra High
Effort multipliers	1.42	1.19	1.00	**0.85**	0.71	n/a

Programmer Capability

Evaluation for this cost driver should be based on the capability of the programmers as a team rather than as individuals. Major factors which should be considered in the rating are ability, efficiency and thoroughness, and the ability to communicate and cooperate.

Table 2.23: PCAP values

PCAP Cost Drivers						
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

The rating for this cost driver is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project team's equivalent level of experience with this type of application.

Table 2.24: APEX values

APEX Cost Drivers						
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

The Post-Architecture model broadens the productivity influence of platform experience by recognizing the importance of understanding the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities.

Table 2.25: PLEX values

PLEX Cost Drivers						
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

This is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem. In addition to experience in the project's programming language, experience on the project's supporting tool set also affects development effort.

Table 2.26: LTEX values

LTEX Cost Drivers						
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

The rating scale for PCON is in terms of the project's annual personnel turnover: team members have been the same since the begin until the end of development.

Table 2.27: PCON values

PCON Cost Drivers						
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

The tool rating ranges from simple edit and code, very low, to integrated life-cycle management tools, very high.

Table 2.28: TOOL values

TOOL Cost Drivers						
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

Determining its cost driver rating involves the assessment and judgement-based averaging of two factors: site collocation (from fully collocated to international distribution) and communication support (from surface mail and some phone access to full interactive multimedia).

Table 2.29: SITE values

SITE Cost Drivers						
SITE Collocation Descriptors	International	Multi-city and multi-company	Multi-city or multi-company	Same city or metro area	Same building or complex	Fully collocated
SITE Communications Descriptors	Some phone, mail	Individual phone, fax	Narrow band email	Wideband electronic communication	Wideband elect. comm., occasional video conf.	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

This rating measures the schedule constraint imposed on the project team developing the software. The ratings are defined in terms of the percentage of schedule stretch-out or acceleration with respect to a nominal schedule for a project requiring a given amount of effort. Accelerated schedules tend to produce more effort in the earlier phases to eliminate risks and refine the architecture, more effort in the later phases to accomplish more testing and documentation in parallel.

Table 2.30: SCED values

SCED Cost Drivers						
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

Cost Driver overall estimation

Here the result of cost drivers' evaluation:

Table 2.31: Cost Drivers overall estimation

Cost Driver	Factor	Value
Required Software Reliability (RELY)	High	1.10
Database size (DATA)	High	1.14
Product complexity (CPLX)	High	1.17
Required Reusability (RUSE)	High	1.07
Documentation match to life-cycle needs (DOCU)	Nominal	1.00
Execution Time Constraint (TIME)	High	1.11
Main storage constraint (STOR)	Nominal	1.00
Platform volatility (PVOL)	Low	0.87
Analyst capability (ACAP)	High	0.85
Programmer capability (PCAP)	Nominal	1.00
Application Experience (APEX)	Very low	1.22
Platform Experience (PLEX)	Low	1.09
Language and Tool Experience (LTEX)	Nominal	1.00
Personnel continuity (PCON)	Very high	0.81
Usage of Software Tools (TOOL)	High	0.90
Multisite development (SITE)	High	0.93
Required development schedule (SCED)	Nominal	1.00
Total (EAF)		1.1618

2.2.3 Effort Equation

COCOMO II provides the following equation to estimate the effort required measured in Person-Months (PM):

$$Effort = A * EAF * KSLOC^{E}$$

Where:

- A is a constant that approximates the productivity in PM / KSLOC
- EAF is the product of all the effort multipliers derived from the Cost Drivers
- E is an aggregation of the five Scale Factors obtained with the following equation:

  $$E = B + 0.01 * \sum_{j=1}^{5} SF_j$$

  where $B = 0.91$.

Given that,

$$A = 2.94$$

$$E = B + 0.01 * 14.64 = 1.0564$$

$$EAF = \prod_{i=1}^{15} EM_i = 1.1618$$

the $Effort$ can be calculated as:

$$Effort = 2.94 * 1.1618 * 6.042^{1.0564} = 22.8411PM \simeq 23PM$$

2.2.4 Schedule Estimation

COCOMO II provides also an equation to estimate the project duration given the parameter previously calculated:

$$Duration = 3.67 * Effort^{F}$$

where F is equal to:

$$F = 0.28 + 0.2 * (E - B)$$

Given that,

$$F = 0.28 + 0.2 * (1.0564 - 0.91) = 0.3093$$

$$Effort = 22.8411PM$$

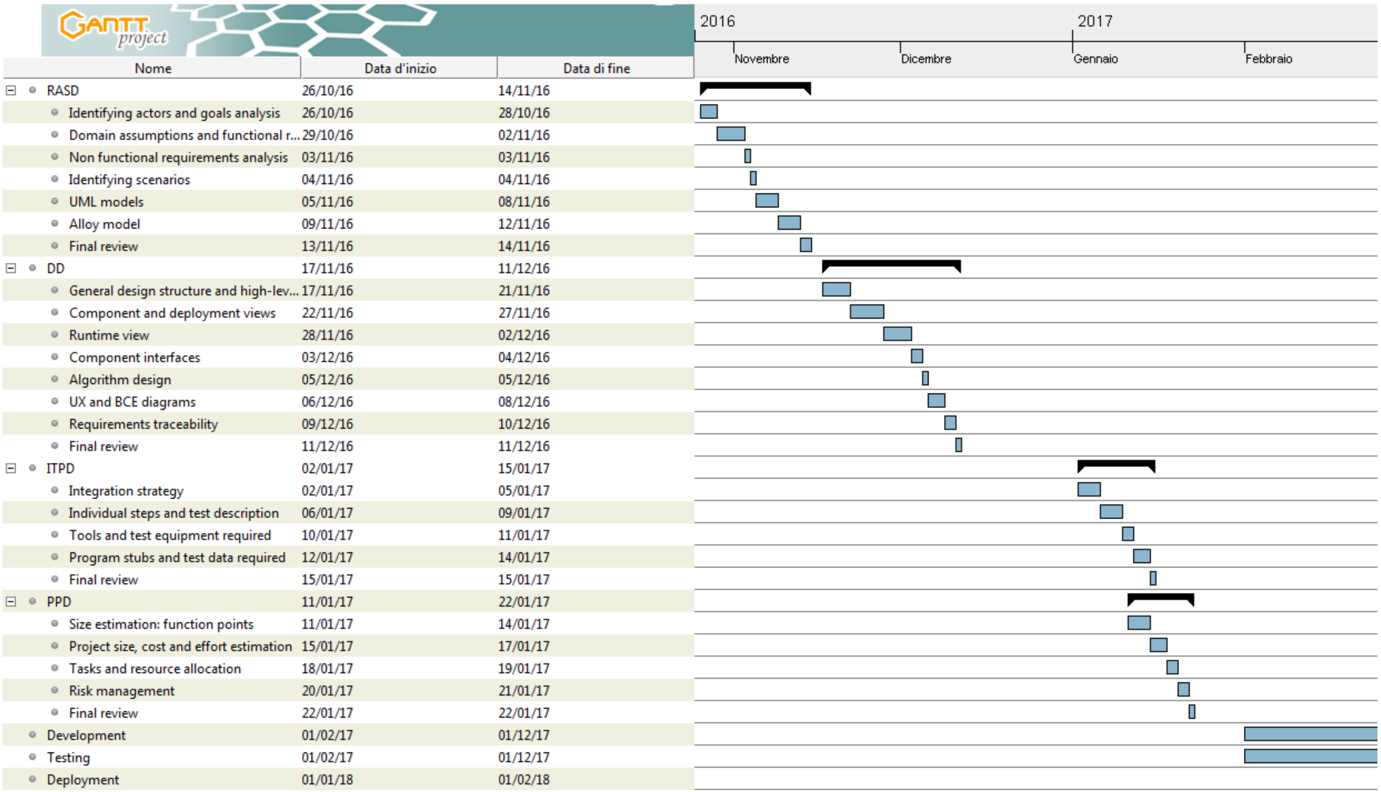
the $Duration$ can be calculated as:

$$Duration = 3.67 * 22.8411^{0.3093} = 9.66\ months$$

Chapter 3

Tasks

In this chapter we are going to provide, thanks to a Gantt chart, a high-level project schedule, defining all main tasks necessary for developing the project. Even if in reality we worked only on RASD, DD, ITPD and PPD, because of the didactic purpose of the project, we decided to provide also the time necessary for developing, testing and finally deploying the system, also using results obtained with COCOMO II analysis.



Nome	Data d'inizio	Data di fine
RASD	26/10/16	14/11/16
Identifying actors and goals analysis	26/10/16	28/10/16
Domain assumptions and functional r...	29/10/16	02/11/16
Non functional requirements analysis	03/11/16	03/11/16
Identifying scenarios	04/11/16	04/11/16
UML models	05/11/16	08/11/16
Alloy model	09/11/16	12/11/16
Final review	13/11/16	14/11/16
DD	17/11/16	11/12/16
General design structure and high-lev...	17/11/16	21/11/16
Component and deployment views	22/11/16	27/11/16
Runtime view	28/11/16	02/12/16
Component interfaces	03/12/16	04/12/16
Algorithm design	05/12/16	05/12/16
UX and BCE diagrams	06/12/16	08/12/16
Requirements traceability	09/12/16	10/12/16
Final review	11/12/16	11/12/16
ITPD	02/01/17	15/01/17
Integration strategy	02/01/17	05/01/17
Individual steps and test description	06/01/17	09/01/17
Tools and test equipment required	10/01/17	11/01/17
Program stubs and test data required	12/01/17	14/01/17
Final review	15/01/17	15/01/17
PPD	11/01/17	22/01/17
Size estimation: function points	11/01/17	14/01/17
Project size, cost and effort estimation	15/01/17	17/01/17
Tasks and resource allocation	18/01/17	19/01/17
Risk management	20/01/17	21/01/17
Final review	22/01/17	22/01/17
Development	01/02/17	01/12/17
Testing	01/02/17	01/12/17
Deployment	01/01/18	01/02/18

Figure 3.1: The picture above shows the first section of Gantt chart representing main tasks of the project.

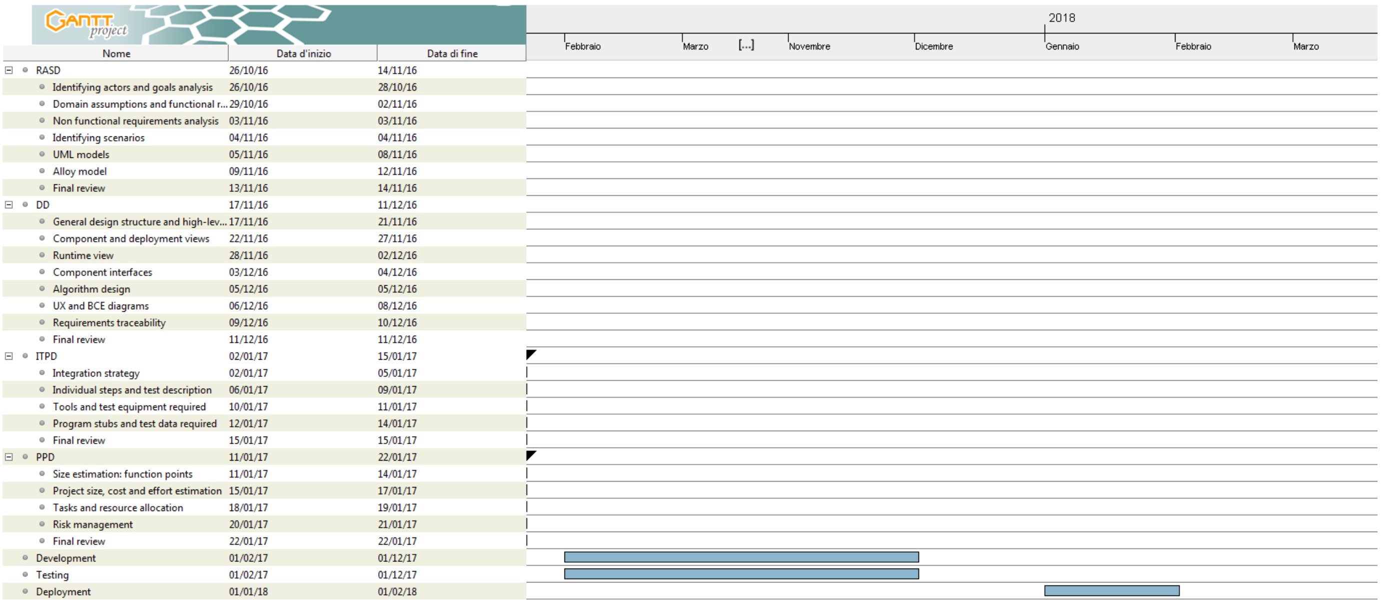


Figure 3.2: The picture above shows the second section of Gantt chart representing main tasks of the project.

Chapter 4

Resource Allocation

In this chapter we are going to provide, thanks to a Gantt chart, a general overview of how resources (in this case the two members of the team) have been assigned to various tasks of the project. For the same reasons explained in chapter 3, we provide the assignment of resources also to development, testing and deployment activities.

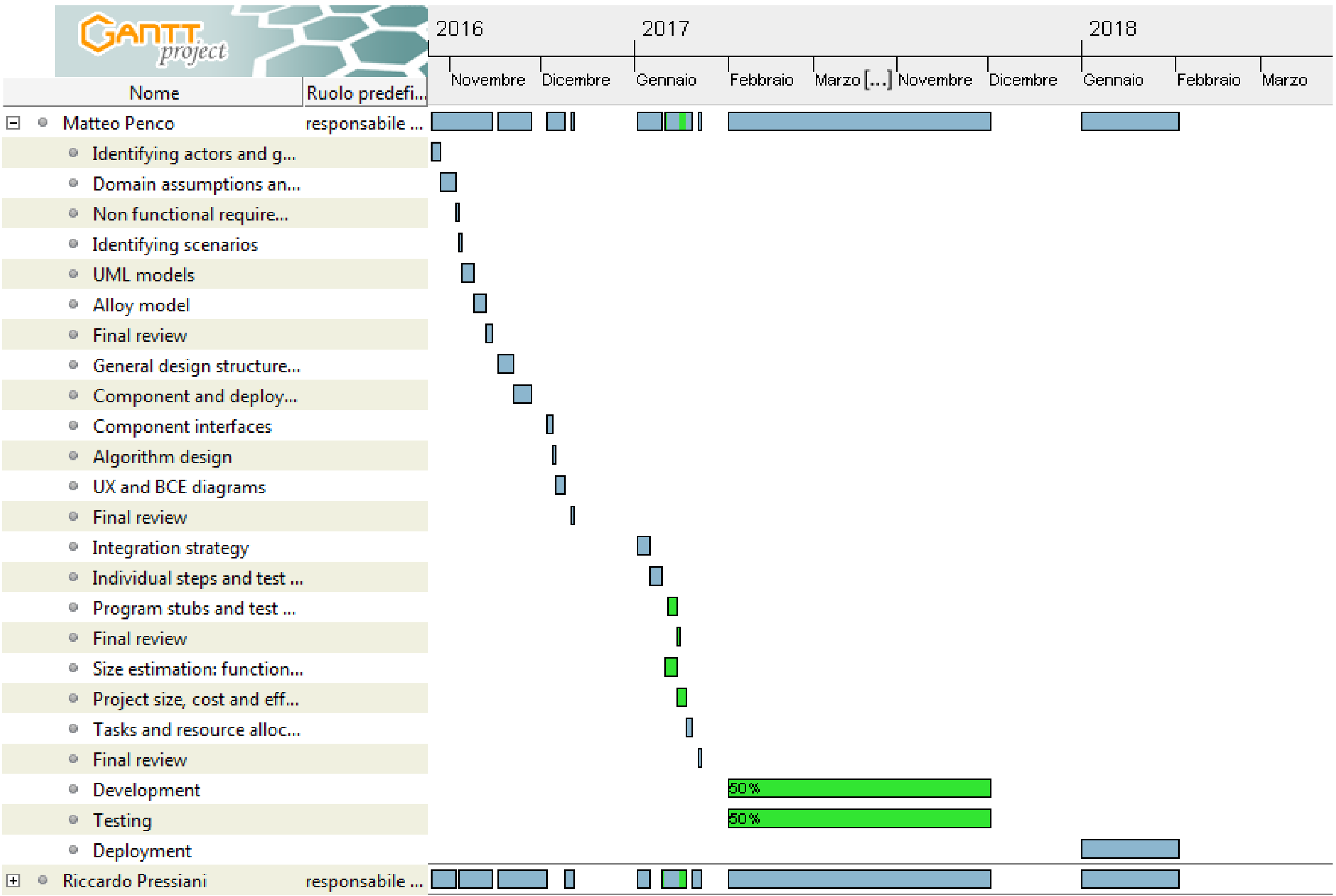


Figure 4.1: The picture above shows the first section of Gantt chart representing resources assigned to various taks of the project.

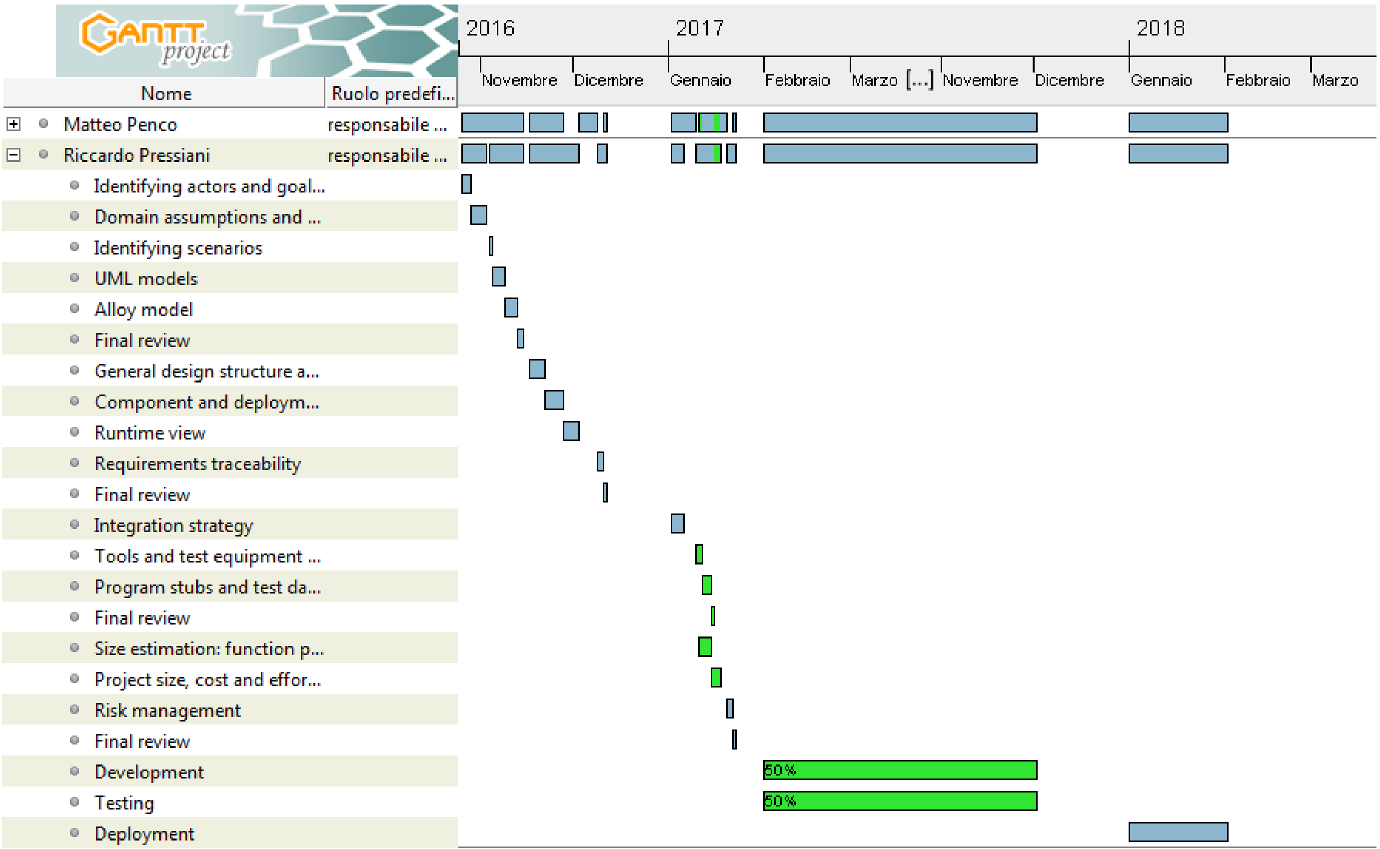


Figure 4.2: The picture above shows the second section of Gantt chart representing resources assigned to various taks of the project.

Chapter 5

Risk Management

In this Chapter, all the known risks related to planning, technical and business issues will be discussed. Along with the risks identified, a possible strategy to overcome them and to reduce the probability will be provided.

First of all, the most important risk that has been identified is the one related to the possible competition with other car sharing companies. Since at this time there are several companies that operates in the same cities where PowerEnJoy is expected to be deployed, it is important to evaluate all the possible scenarios related to this risk. The strategy to overcome this issue needs to be discussed with the marketing team. Well-planned marketing campaigns, interesting prices and services with respect to other companies and simple processes to learn how to use the service can be good strategies to adopt.

Usually, projects that involves a service for the citizens are sponsored by local governments. It is fundamental to organize meeting with the local authorities in order to make them understand the importance and the benefits that PowerEnJoy would bring to the local communities. It is important to collect guarantees from the local governments about the expected economic investment in PowerEnJoy in order to deliver a complete and well designed product and to avoid wrong budget planning.

Another risk is the one related to the requirements volatility. A thorough requirement elicitation phase is fundamental to collect all the projects specifications before taking any decision about the design or the implementation. PowerEnJoy is an application that can be expanded and integrated with multiple features. For this reason, it is important to come to an agreement with all the stakeholders before any development activities start.

Since PowerEnJoy relies on multiple external service, the malfunction, the modification or the closure of one of those would be a serious issue to deal with. The three main external services integrated in PowerEnJoy are the notification, payments and recovery service. Every services mentioned is fundamental for the correct functioning of the application. For this reason, a thorough investigation must be done in order to select the most reliable and known companies in each of those sectors.

Although the overall process to take advantage of the PowerEnJoy services is intrinsically quite simple, the QA team must assure that the users will be able to easily learn how to use PowerEnJoy. If the users will face difficulties while approaching the app the first time, the effect could be catastrophic. The strategies that can be exploited to overcome this kind of risk are tests focused on the users, such as A/B and beta testing of the mobile and web application along with the interfaces of the onboard computers.

PowerEnJoy must guarantee a 24/7 service. The possible downtimes that could happen if the infrastructure supporting PowerEnJoy would not be available represent a critical risk for the whole system. Important economic losses are expected if long downtimes happen. The strategy that can be applied is to make a thorough analysis of all the possible cloud providers in order to select the most suitable one for the system to be developed and deployed.

Finally, the personnel turnover is a chance that must be considered. Serious delays of the system delivery must be expected if new developers join the project in the middle of the development process. As an example, a certain amount of time must be estimate to let the new employees learn the technologies that are used and how the already developed component works. To overcome this risk the personnel selection phase must be conduct with the right attention and the code base must be fully and high detailed documented.

List of Figures

List of Tables

Bibliography

[1] Apple Inc. iOS. Available at http://www.apple.com/ios/ (November 26, 2016).

[2] Google Inc. Android. Available at https://www.android.com/ (November 26, 2016).

[3] L. Mottola and E. Di Nitto. *Assignments AA 2016-2017*. Politecnico di Milano.

[4] M. Penco and R. Pressiani. PowerEnjoy - Requirement Analysis and Specification Document. Available at https://github.com/rpressiani/se2-project-2016/blob/master/releases/RASD.pdf (November 27, 2016).

[5] M. Penco and R. Pressiani. PowerEnjoy - Design Document. Available at https://github.com/rpressiani/se2-project-2016/blob/master/releases/DD.pdf (January 03, 2017).

[6] M. Penco and R. Pressiani. PowerEnjoy - Integration Test Plan Document. Available at https://github.com/rpressiani/se2-project-2016/blob/master/releases/ITPD.pdf (January 18, 2017).

[7] QSM Quantitative Software Management. Function Point Languages Table. Available at http://www.qsm.com/resources/function-point-languages-table (January 20, 2017).